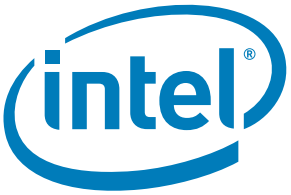
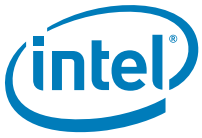


Intel® Embedded Graphics Drivers and Video BIOS v8.0

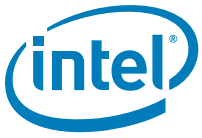
User's Guide

October 2007

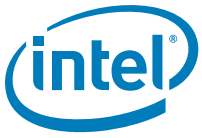
Document Number: 274041-013US



Contents



Figures

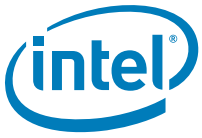


Tables

Revision History

This document may have been updated since the release shown below. See http://www.intel.com/design/intarch/manuals/274041.htm for the most recent version.

Date	Revision	Description
October 2007	013	Updated for use with Version 8.0 of the product. Change bars indicate areas of change.
June 2007	012	Updated for use with Version 7.0 of the product.
December 2006	011	Updated for use with Version 6.1 of the product.
September 2006	010	Updated for use with Version 6.0 of the product, including support for the Intel® Q965 and Damn Small Linux*.
June 2006	009	Updated for use with Version 5.1 of the product, including support for the Texas Instruments TFP410* DVO encoder, Microsoft Windows Embedded for Point of Service (WEPOS)* operating system, and SuSE 10.
February 2006	008	Updated for use with Version 5.0 of the product, including support for the Intel® 852GM, Intel® 945G, and Intel® 945GM chipsets, the Silicon Image SiI 1362* and SiI 1364* SDVO transmitters, and External PCI as a Primary graphics adaptor.
October 2005	007	Updated for use with Version 4.1 of the product.
June 2005	006	Updated for use with Version 4.0 of the product, including support for the Intel® 915GV and Intel® 915GM chipsets, the Chrontel CH7307* and Chrontel CH7308* SDVO transmitters, and Advanced EDID Configuration.
May 2005	005	Updated for use with Version 3.4 of the product, including use of the enhanced Video BIOS, Windows* installer/uninstaller, runtime configuration GUIs, and display discovery feature.
July 2004	004	Updated for use with Version 3.2 of the product, including use of the dynamic port driver feature.
May 2004	003	Updated for usage with version 3.1 of the product, including details on PCF format and usage, Universal INF format, and updates to the User Build System.
February 2004	002	Updated chipset support to reflect current Embedded IA32 roadmap.
February 2004	001	Initial Release



1.0 Introduction

The Intel® Embedded Graphics Drivers (IEGD) comprise a suite of multi-platform graphics drivers designed to meet the requirements of embedded applications. Featuring Intel® Dynamic Display Configuration Technology (DDCT), the drivers run on the following Embedded Intel® Architecture (eIA) chipsets:

- Intel® Q35 Express chipset
- Mobile Intel® GME965 Express chipset
- Intel® Q965 Express chipset
- Mobile Intel® 945GME Express chipset
- Intel® 945G Express chipset
- Intel® 915GV Express chipset
- Mobile Intel® 915GME Express chipset
- Mobile Intel® 910GMLE Express chipset
- Intel® 855GME chipset
- Intel® 852GME chipset
- Intel® 852GM chipset

Note: If you need support for a chipset that is not listed above but is in the same family as those listed, please contact your Intel representative.

The IEGD supports four types of display devices:

- Analog CRT
- LVDS flat panels
- TMDS DVI displays
- TV Output

The IEGD is designed to work with fixed-function systems, such as Point-of-Sale (POS) devices, ATM machines, gaming devices, etc. It can be configured to work with various hardware and software systems and supports both Microsoft Windows* and Linux* operating systems, including embedded versions of these operating systems.

The Intel Embedded Graphics Suite consists of both the IEGD and a Video BIOS (VBIOS) component. These two components are configurable and work together to provide a wide range of features. This document provides information on configuring and using both the IEGD and the VBIOS.

The IEGD provides the following features:

- Enhanced VBIOS support
- Dynamic Port Drivers
- Support for Dual Independent Head (DIH) displays
- Support of a Universal INF file

- EDID and EDID-less display support
- Display discovery and initialization
- Direct 3D* support
- Installer/Uninstaller GUI for Microsoft Windows
- Runtime configuration GUI for Microsoft Windows and Linux

1.1 Purpose

This manual provides information on both firmware and software, providing hardware design considerations, installation requirements, and static configuration options.

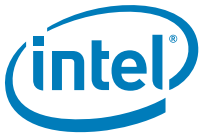
1.2 Intended Audience

This document is targeted at all platform and system developers who need to interface with the graphics subsystem. This includes, but is not limited to: platform designers, system BIOS developers, system integrators, original equipment manufacturers, system control application developers, as well as end users.

1.3 Related Documents

The following documents provide additional information on the hardware supported by the IEGD.

- *Intel® Embedded Graphics Drivers for Embedded Intel® Architecture-based Chipsets Product Brief*
  (Document Number: 315587)
- *Intel® 35 Express Chipset Family Datasheet*
  (Document Number: TBD)
- *Mobile Intel® 965 Express Chipset Family Datasheet*
  (Document Number: 316273)
- *Intel® 965 Express Chipset Family Datasheet*
  (Document Number: 313053)
- *Mobile Intel® 945GM/PM/GMS Express Chipset Datasheet*
  (Document Number: 309219)
- *Mobile Intel® 915PM/GM/GMS and 910GML Express Chipset Datasheet*
  (Document Number: 305264)
- *Intel® 915G/915GV/915P Express Chipset Datasheet*
  (Document Number: 304467)
- *Intel® 855GM/GME Chipset Graphics and Memory Controller Hub (GMCH) Datasheet*
  (Document Number: 252615)
- *Intel® 852GME Chipset GMCH and Intel® 852GM Chipset MCH Datasheet*
  (Document Number: 253027)
- *Intel® I/O Controller Hub 6 (ICH6) Family Datasheet*
  (Document Number: 301473)
- *Integrated Dual Independent Display on Intel® Digital Security Surveillance Multifunction Platforms Application Brief*
- *Display Panel Debugging with the Intel Graphics Memory Controller Hub*
  (Document Number: 305964)



- *VESA BIOS Extensions/Display Data Channel Standard,* available at the following Web address:

  http://www.vesa.org/public/VBE/VBEDDC11.PDF

  This document provides information on the 4F VBE functions, which are supported by the Intel embedded Video BIOS.
- *VESA BIOS Extension (VBE) Core Functions Standard Version 3.0,* available at the following Web address:

  http://www.vesa.org/public/VBE/vbe3.pdf

  Contains information on the VESA BIOS Extension (VBE) specification for standard software access to graphics display controllers that support resolutions, color depths, and framebuffer organizations beyond the VGA hardware standard.

1.4 Conventions

The following conventions are used throughout this document.

Boldface	Represents text that you type and text that appears on a screen.
Italics	Introduces new terms and titles of documents.
`Courier New`	Identifies the names of files, executable program names, and text that appears in a file.
Angle Brackets (<>)	Encloses variable values in syntax or value ranges that you must replace with actual values.
Vertical Bar (|)	Used to separate choices (for example, TRUE | FALSE)

1.5 New Features for Version 8.0

Table 1 presents new IEGD features and capabilities.

Table 1. Intel® Embedded Graphics Drivers and Video BIOS 8.0 New Features

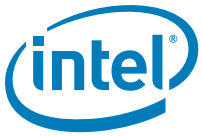
New Features
D3D-M for Windows CE 6.0 • Supported platforms: Q963, Q965 and GM(E)965. IEGD 8.0 will not cover D3D-M support for Intel® Q35 Express chipsets. • Covers vertex processing emulation on all platforms, which includes 852, 855, 915GV, 915GM(E), 945G, 945GM(E), Q963, Q965, GM(E)965. Vertex Processing will not cover Q35.
Q35 Gen3.5 Core with 2D • 2D • Overlay and 2nd overlay • Display resolution/ timings • Dynamic mode change • Rendered Scaling • Display configuration - Single, Twin, Clone, Extended, Dual Independent Display • Rotation and inverted display • IEGD 8.0 does not cover Q35 on D3D, OGL, D3D-M and DSL.
Q35 Legacy VBIOS Support • VGA modes • VESA modes • Non-standard modes • EDID and EDID-less display • Display ID

Table 1. Intel® Embedded Graphics Drivers and Video BIOS 8.0 New Features

New Features
Fedora Core 6 • Support on all platforms includes Q35
Fedora Core 7 • Support on all platforms includes Q35
SUSE Linux Enterprise Server 10 (SU10SP1) Suse 10+ • Support on all platforms includes Q35 • Kernel 2.6.16.46-0.12 and X.org 6.9 with service pack 1. **Note:** Use the IEGD X.org 7.0 on systems that use either X.org 6.9 or 7.0.
IEGD Kernel Module (IKM) • AGPGart in the form of a kernel module eliminates kernel patching and recompilation • Enables other distributions not directly supported by IEGD to use IEGD • Source code provided • Uninstaller included for uninstalling the related module from the kernel. Mostly in a script form to remove old agpgart module and replaced with the new one that supports IEGD.
Second Overlay for Gen4, includes Q35 • Enabled for Q963, Q965, GM(E)965, and Q35. • Continue to support existing chipsets (852, 855, 915GV, 915GM(E), 910GML(E), 945G and 945GM(E)) that already support this feature • Support across all OSes in IEGD 8.0 • Enabled GUI for 2nd overlay color correction.
Text Tuning on SDVO Port • This feature applies to all IEGD 8.0 supported transmitter regardless of DVI or TV or CRT output. • GUI allows text tuning control • CED enables default setting of text tuning
CRT on Second Display on SDVO Port (VGA Bypass) • This feature is supported on CH7021 and CH7317 transmitters
Silent Install Support • Enables usage model of remote driver upgrade • Allows options for installation without input through special parameters
CED Support for Q35
New Transmitter Support • VGA bypass with CH7021 and CH7317 • Preliminary HDMI support with CH7315 — No HDCP or Audio sync • DVI with CH7319 and CH7320 support — No support for HDCP with CH7319

This release also contains resolutions for errata. For details on errata, including status information, refer to the specification update located at the following Web address:

http://www.intel.com/design/intarch/specupdt/309380.htm



1.6 Acronyms and Terminology

Table 2 lists the acronyms and terminology used throughout this document.

Table 2. Acronyms and Terminology (Sheet 1 of 3)

Term	Description
ADD Card	APG Digital Display. An adapter card that can be inserted into the AGP port of Intel chipset family-based systems. ADD cards allow configurations for TV-out, LVDS, and TMDS output (i.e., televisions, digital displays, and flat panel displays).
AGP	Accelerated Graphics Port. An interface specification that allows 3-D graphics to be displayed smoothly and quickly on a display device.
AIM	Add In Module.
API	Application Programming Interface.
BDA	BIOS Data Area. A storage area that contains information about the current state of a display, including mode number, number of columns, cursor position, etc.
BIOS	Basic Input/Output System. The IEGD interacts with two BIOS systems: system BIOS and Video BIOS (VBIOS). VBIOS is a component of the system BIOS.
CED	Configuration EDitor. Graphical pre-installation utility allows easy creation of consolidated driver installation packages for Windows, Windows CE, Linux, and VBIOS across numerous platforms and display combinations.
Clone Display Configuration	A type of display configuration that drives two display devices, each displaying the same content, but can have different resolutions and (independent) timings. Compare Twin Display Configuration and DIH Display Configuration.
DDCT	Intel® Dynamic Display Configuration Technology
DirectDraw*	A component of the DirectX* Graphics API in Microsoft Windows.
DIH Display Configuration	Dual Independent Head. A type of display configuration that supports two displays with different content on each display device. The IEGD supports Extended mode for Microsoft Windows systems and Xinerama for Linux systems.
DTD	Detailed Timing Descriptor. A set of timing values used for EDID-less devices.
DVI	Digital Video Interface.
DVO	Digital Video Out. A port on the GMCH that allows connection to a digital transmitter, either an ADD card or on-board transmitter, and permits connections to various digital devices, such as TVs, LVDS flat panel displays, and TMDS devices. The GMCH provides up to three DVO ports named DVOA, DVOB, and DVOC.
EBDA	Extended BIOS Data Area. An interface that allows the system BIOS and Option ROMs to request access to additional memory.
EDID	Extended Display Identification Data. A VESA standard that allows the display device to send identification and capabilities information to the IEGD. IEGD reads all EDID data, including resolution and timing data, from the display, thus negating the need for configuring DTD data for the device.
EDID-less	A display that does not have the capability to send identification and timing information to the driver and requires DTD information to be defined in the driver.
eIA	Embedded Intel® Architecture.
EMI	Electromagnetic Interference.
Extended Clone Mode	A feature that allows you to have different sized displays in Clone mode.
Framebuffer	A region of physical memory used to store and render graphics to a display.

Table 2. Acronyms and Terminology (Continued) (Sheet 2 of 3)

Term	Description
Gang DVO	A type of configuration for the NS387R transmitter that gets data from 2 DVO ports (DVO-B and DVO-C) and sends it to a single display.
GEN3	Napa Graphics Core in 910/915 family chipset.
GEN3.5	Napa+ Graphics Core in 945 family chipset.
GEN4	Graphics Core in 965 family chipset.
GDI	Graphics Device Interface. A low-level API used with Microsoft Windows operating systems.
GMA	Intel Graphics Media Accelerator. Refers to both the graphic hardware in Intel chipsets as well as the desktop/mobile driver. The GMA driver is not intended for use in embedded applications.
GMCH	Graphics and Memory Controller Hub.
GMS	Graphics Mode Select (stolen memory).
HAL	Hardware Abstraction Layer. An API that allows access to the Intel® 855GME and 852GME chipsets.
IAL	Interface Abstraction Layer. An API that allows access to graphics interfaces including the GDI, DirectDraw*, XFree86*.
IEGD	Intel® Embedded Graphics Drivers
IEGS	Intel® Embedded Graphics Suite. Runtime graphics driver plus a VBIOS component.
INF file	A standard Microsoft Windows text file, referred to as an information file, used by Microsoft Windows to provide information to the driver. The default .inf file for the IEGD is `iegd.inf`. You can create customized parameters using the CED utility.
LVDS	Low Voltage Differential Signaling. Used with flat panel displays, such as a laptop computer display.
Linux/XFree86	Open Source for XWindows* used on Linux systems.
NTSC	National Television Standards Committee. A TV standard used in North and Central America and in Japan.
OAL	Operating System Abstraction Layer. An API that provides access to operating systems, including Microsoft Windows and Linux.
Option ROM	Code which is integrated with the system BIOS and resides on a flash chip on the motherboard. The Intel Embedded Video BIOS is an example of an option ROM.
PAL	Phase Alternating Lines. A TV standard used in Europe, South America, Africa, and Australia.
PCI	Peripheral Component Interface.
Port Driver	A driver used with the DVO interfaces of the Graphics and Memory Controller Hub (GMCH).
POST	Power On Self Test.
Reserved Memory	A region of physical memory in a Windows CE* system set aside for BIOS, VBIOS, and Graphics Driver operations. Reserved memory can be configured to be used by the operating system and other applications when not in use by the BIOS.
sDVO	Serial Digital Video Output.
Single Display Configuration	A type of display configuration that supports one and only one display device.
Stolen Memory	A region of physical memory (RAM) set aside by the system BIOS for input and output operations. The amount of stolen memory is configurable. Stolen memory is not accessible to the operating system or applications.
System BIOS	The standard BIOS used for basic input and output operations on PCs.

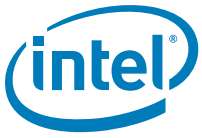


Table 2. Acronyms and Terminology (Continued) (Sheet 3 of 3)

Term	Description
TMDS	Transitioned Minimized Differential Signaling. Used with DVI displays, such as plasma TVs.
TOM	Top Of Memory.
TSR	Terminate and Stay Resident. A program that is loaded and executes in RAM, but when it terminates, the program stays resident in memory and can be executed again immediately without being reloaded into memory.
Twin Display Configuration	A type of display configuration that supports two display devices each of which has the same content, resolution, and timings. Compare Clone Display Configuration.
UBS	User Build System. A process for building a VBIOS.
VBIOS	Video Basic Input Output System. A component of system BIOS that drives graphics input and output.
VESA	Video Electronics Standards Organization.
VGA	Video Graphics Array. A graphics display standard developed by IBM* that uses analog signals rather than digital signals.

1.7 Downloading the IEGD and Video BIOS

The IEGD and the Video BIOS (VBIOS) are available via the Download Drivers link (under Related Information) at http://www.intel.com/go/iegd or directly via the Intel Embedded Graphics Drivers Downloads page, where the following is available:

- Intel Embedded Graphics Driver Configuration Editor (CED) release
  - — includes the IEGD drivers for VBIOS, Linux, and all Windows, plus an online help system

Note: The Embedded Video BIOS version 8.0 is recommended for use with each of the graphics drivers in most cases. The Legacy Video BIOS version 3.2.1 may be used in Intel 845G and 855GME chipset-based systems, but it is recommended that you use the most current IEGD VBIOS. Click the following link to see the FAQ page for details on the differences of these versions.

http://www.intel.com/design/intarch/swsup/graphics_faq.htm

Once you have downloaded, installed, and run CED, you can configure and customize the drivers and VBIOS following the procedures in this document. Once they have been configured, you can integrate the VBIOS with the system BIOS ROM and install the IEGD on your operating system.

1.8 Manually Configuring and Installing the IEGD and Video BIOS

It is recommended to use CED to configure and install the IEGD and VBIOS. However, if you prefer to configure and install manually, the following procedure outlines the process.

1. Download the VBIOS and the IEGD software.
2. Unzip the files. For the Video BIOS files, unzip them to a folder near the top of the filesystem root as there is character limitation in DOS. Also note that the User Build System configuration tools require relative locations of the subdirectories. Once you have unzipped the files, do not move or rename any of the folders or files.
3. Configure the VBIOS. If you are using the legacy VBIOS, update the VBIOS Configuration File (`def_eg.txt`). Please see Chapter 8.0, "Legacy VBIOS" for instructions on configuring the legacy VBIOS.

   If you are using the new VBIOS, use CED. Please see Section 3.1, "IEGD Configuration Editor (CED)" on page 23 and the CED Help system.
4. Configure the System BIOS. For instructions, see Section 3.3, "System BIOS Settings" on page 24.
5. Build the VBIOS Option ROM using the User Build System process described in Chapter 4.0, "VBIOS". If you are using the legacy VBIOS, refer to Chapter 8.0, "Legacy VBIOS".
6. Integrate the VBIOS Option ROM with the system's firmware image using tools provided by your system BIOS vendor. For example, if you are using the American Megatrends Inc.* (AMI) system BIOS, use `mmtools.exe`.
7. If required, configure the Driver by updating the config file. Please see Chapter 3.0, "Platform Configuration".
8. Install the IEGD on your operating system. If you are installing to a Microsoft Windows system, use the Microsoft Windows installer program described in Section 5.6, "Viewing and Changing the Driver Configuration from Microsoft Windows" on page 72.

   If you are installing to a Linux operating system, follow the installation procedure described in Chapter 7.0, "Installing and Configuring Linux Drivers".

After the IEGD is installed, you can make runtime configuration changes by using the Microsoft Windows or Linux runtime configuration GUI. See Section 5.6, "Viewing and Changing the Driver Configuration from Microsoft Windows" on page 72 for information on using the Microsoft Windows Runtime Configuration GUI and Section 7.7.1, "Runtime Configuration GUI (iegdgui)" on page 129 for using the Linux Configuration GUI. In general, changes you make to the IEGD during runtime take effect immediately and there is no need to reboot the operating system.

2.0 Architectural Overview

2.1 Introduction

The Intel Embedded Graphics Suite (IEGS) is composed of a runtime graphics driver and a Video BIOS (VBIOS) firmware component. (See Figure 1 through Figure 3.) Both the driver and VBIOS control the GMCH to perform display and render operations. The VBIOS is predominantly leveraged by System BIOS during system boot but is also used at runtime by the driver to handle full-screen text mode on Microsoft Windows* operating systems.

Figure 1. Intel Embedded Graphics Suite

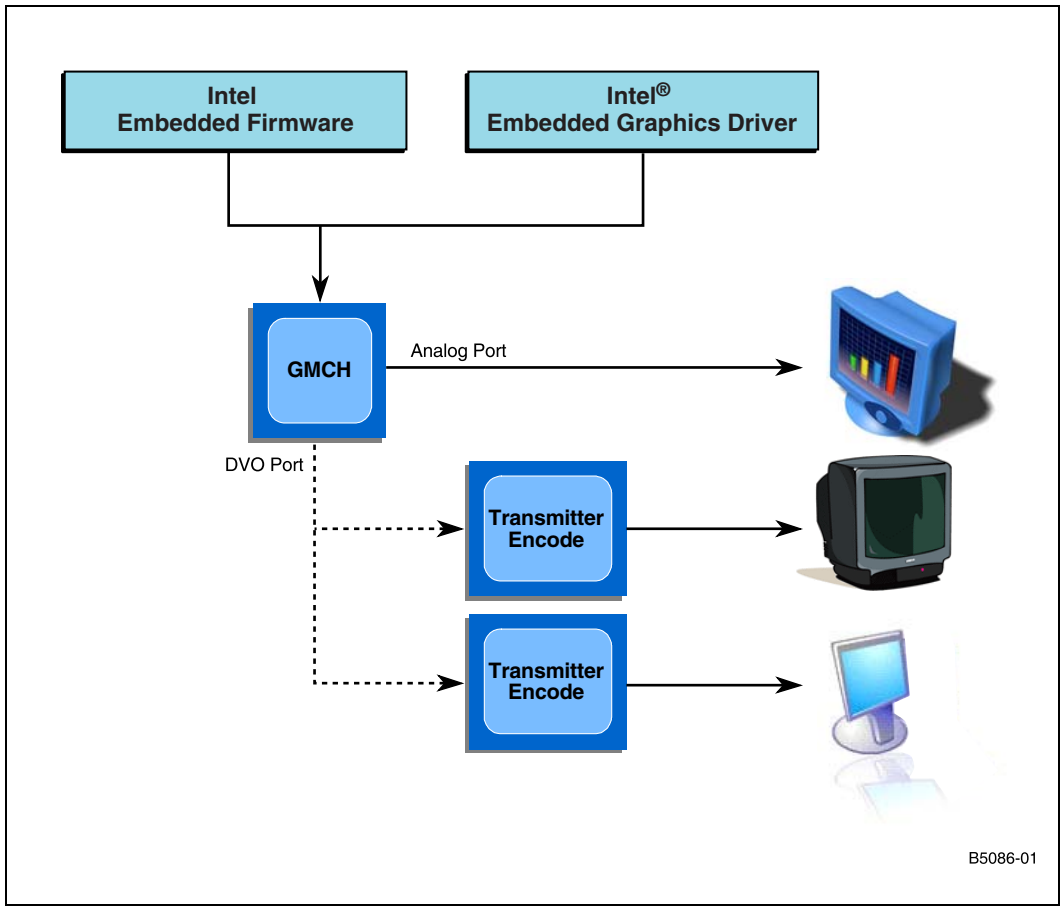


Figure 2. Graphics Driver Architecture

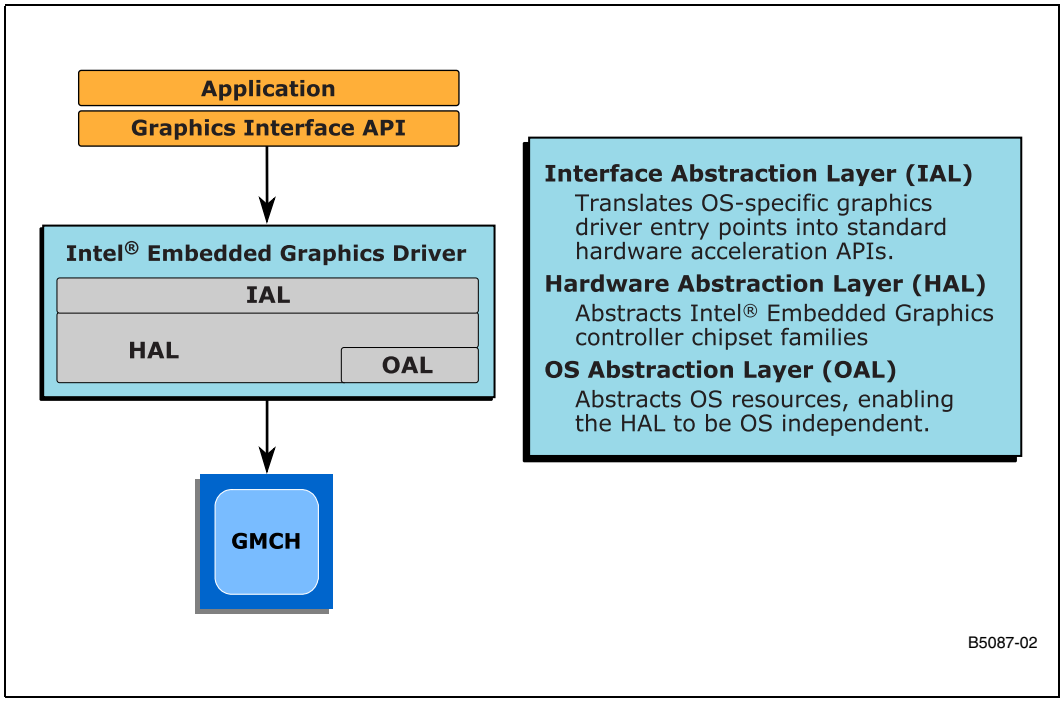
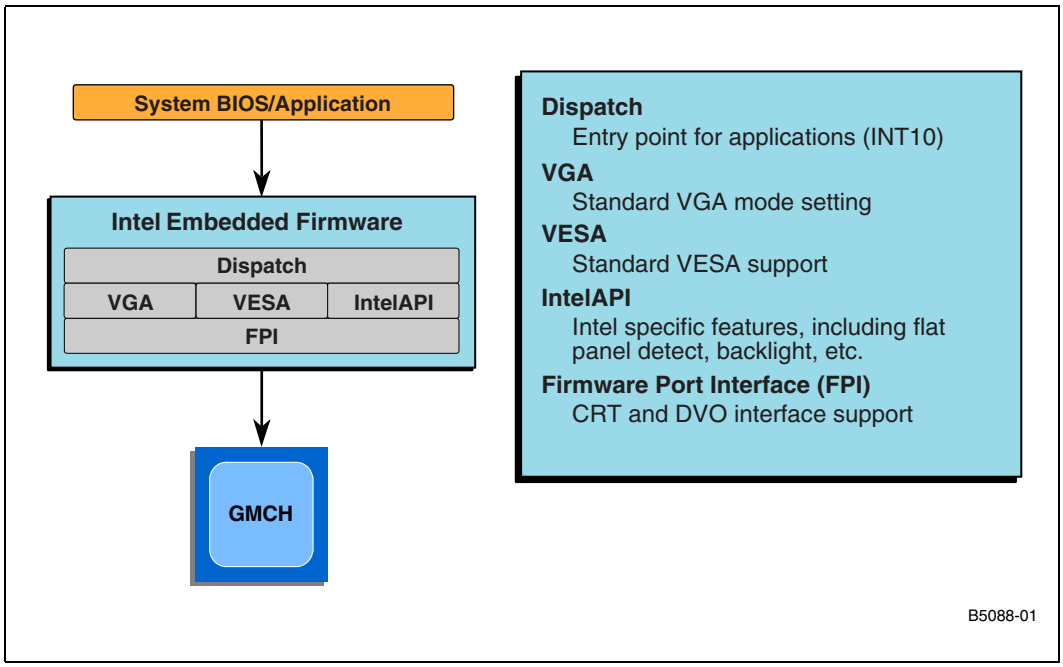


Figure 3. Firmware Architecture



2.1.1 Display Options

The following section describes the types of displays and configurations supported by the Intel Embedded Graphics Driver.

2.1.1.1 Types of Displays

The table below lists the types of displays supported by the IEGD.

Table 3. Types of Displays

Display	Description
CRT	Analog CRT, natively supported with RGB signaling.
Flat Panel	TMDS and LVDS compliant flat panels are supported with the use of an external transmitter via a DVO or sDVO port. Integrated LVDS flat panels are also natively supported on the Intel® 852GME, Intel® 855GM, Mobile Intel® 910GMLE, Mobile Intel® 915GME Express, Mobile Intel® 945GME, and Mobile Intel® GME965 chipsets.
TV	TV-out is supported via an external encoder DVO or sDVO port. ***Note:*** TV-Out is enabled via supported internal capability, or external transmitters DVO/ sDVO (where available).

2.1.1.2 Display Configuration

IEGD supports driving two displays simultaneously. Several configurations are supported, dependent on operating system and chipset. The various display configuration are described in Table 4.

Table 4. Display Configuration Definitions

Display Configuration Mode	Description
Single	Normal desktop configuration, single monitor
Twin	Two displays, same content, single resolution, same timings
Clone	Two displays, same content, different resolutions, independent timings
Extended	Two displays, continuous content
DIH	Dual Independent Head. Two displays, different content, independent resolutions

Table 5 summarizes which display configurations are supported by Intel chipsets.

Table 5. Supported Display Configurations

Chipset	Operating System		
	Windows XP*	Windows CE*	Linux
Intel® Q35 Intel® GME965, Intel® Q965, Intel® 945GM, Intel® 945G, Intel® 915GME, Intel® 910GMLE	Single, Twin, Clone, Extended	Single, Twin, Clone	Single, Twin, Clone, DIH
Intel® 915GV	Single, Twin, Clone	Single, Twin, Clone	Single, Twin, Clone
Intel® 852GME, Intel® 852GM, Intel® 855GME	Single, Twin, Clone, Extended	Single, Twin, Clone	Single, Twin, Clone, DIH

Twin and Clone modes are supported by IEGD through custom APIs. In contrast, Extended and DIH are supported natively by both Microsoft Windows and Linux (XFree86 and X.org*).

2.2 Features

The following sections describes major features supported by IEGD.

2.2.1 Chipsets Supported

Table 6 lists IEGS-supported chipsets.

Table 6. Chipsets Supported by the Intel Embedded Graphics Suite

Chipset	IEGD Legacy VBIOS Support	IEGD VBIOS Support	IEGD Support
Intel® Q35	No	Yes	Yes
Intel® GME965, Intel® Q965	No	Yes	Yes
Intel® 945G Intel 945GME	No No	Yes Yes	Yes Yes
Intel® 915GV Intel® 915GME	No No	Yes Yes	Yes Yes
Intel® 910GMLE	No	Yes	Yes
Intel® 855GME	Yes	Yes	Yes
Intel® 852GME Intel® 852GM	Yes No	Yes Yes	Yes Yes

All supported chipsets provide support for a single analog output for CRTs. In addition, digital monitors, flat panels and TVs are supported through the GMCH DVO and sDVO interface.

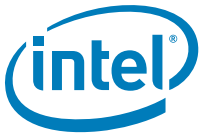
2.2.2 OS and API Support

The IEGD and Video BIOS support the following operating systems and APIs. For OpenGL APIs, see Appendix D, “Intel® OpenGL APIs”.

- Linux XFree86 and X.org
- Damn Small Linux
- Microsoft Windows XP, Windows XP Embedded*
  - — DirectX* 8.1 (DirectDraw* and Direct3D*)
  - — DirectX 9 (DirectDraw and Direct3D)
- Microsoft Windows CE 5.0 and 6.0

Note: The following features are NOT supported in IEGD 8.0:

- Microsoft Vista 2D + 3D
- Vista DirectX 9.0L, DirectX 10.0 (Combine with MS Vista 2D + 3D)



2.2.3 EDID-Less Configuration

EDID-less support is the ability to run a display panel that does not have display timing information within the panel. Therefore, the user has to provide the display timing information to the graphics drivers. For the IEGD, this must be done through:

- VBIOS User Build System for the VBIOS
- Configuration file for the graphics drivers.

This document describes only the necessary edits to the configuration files that are required to implement the graphics driver and VBIOS, and not specific settings for EDID-less panel configuration. Please refer to the manufacturer's specifications for the DTD settings to use for your EDID-less panels.

2.2.3.1 EDID-Less Panel Type Detection

The Intel Embedded Graphics Suite supports EDID-less displays that do not export timing modes. This is accomplished by allowing configuration of a Detailed Timing Descriptor (DTD), and associating that DTD with a specific display port. The IEGS provides further flexibility in allowing numerous DTDs to be defined and having the selection of the DTD be configurable though selection of Configuration IDs. The selection of the Configuration ID can be done from the System BIOS, as long as it supports the Intel 5F40h function and passes the appropriate Configuration ID to the VBIOS. The VBIOS in turn notifies the Graphics Driver of which Configuration ID is active. This is not required however, but the VBIOS and/or Graphics Driver require the Configuration ID to be set prior to installation.

2.2.4 DVO Devices

The IEGD supports many third-party digital transmitters connected to the DVO ports of the GMCH. The driver code that supports each of these devices is abstracted and is a separate driver called a port driver. Port drivers can be dynamically loaded at the time IEGD is initialized, and IEGD can be configured to allow any number of these port drivers to be loaded. By default, all the port drivers for the devices listed in the following table as Included in Release Package will be loaded by default if the corresponding transmitter is detected. If a port driver is not specified in the configuration before installation, that device will not be detected, and the port driver will not be loaded. The configuration can be modified before installation to prevent certain port drivers from being loaded or to include additional port drivers to load.

Table 7. DVO/SDVO Devices Supported (Sheet 1 of 2)

Device	Legacy VBIOS Support	VBIOS Support	Graphics Driver Support
Internal LVDS	√	√	√
Internal TV Out			√
Chrontel CH7009/CH7010*	√	√	√
Chrontel CH7017*		√	√
Chrontel CH7021*		√	√
Chrontel CH7305*		√	√
Chrontel CH7301*		√	√
Chrontel CH7307* (sDVO)		√	√
Chrontel CH7308* (sDVO)		√	√
Chrontel CH7317*		√	√

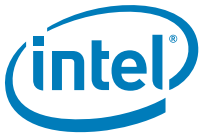
Table 7. DVO/SDVO Devices Supported (Continued) (Sheet 2 of 2)

Chrontel CH7315/CH7319/CH7320*		√	√
Focus FS453/FS454*		√	√
National Semiconductor NS2501*	√	√	√
National Semiconductor NS387R*		√	√
Silicon Image SiI 164*	√	√	√
Silicon Image SiI 1362* (sDVO)		√	√
Silicon Image SiI 1364* (sDVO)		√	√
Texas Instruments TFP410*		√	√
THine Th164*	√	√	√

2.2.5 Rotation

Rotation is the ability to rotate the display for the Intel Embedded Graphics Driver. Rotation support includes 0°, 90°, 180°, 270°. Rotation is supported only on the following chipsets using Windows XP and Linux:

- Intel® Q35 Express chipset
- Mobile Intel® GME965 Express chipset
- Intel® Q965 Express chipset
- Mobile Intel® 945GME Express chipset
- Mobile Intel® 945GM Express chipset
- Intel® 945G Express chipset
- Intel® 915GV Express chipset
- Mobile Intel® 915GME Express chipset
- Mobile Intel® 910GMLE Express chipset
- Intel® 855GME chipset
- Intel® 852GME chipset
- Intel® 852GM chipset



3.0 Platform Configuration

3.1 IEGD Configuration Editor (CED)

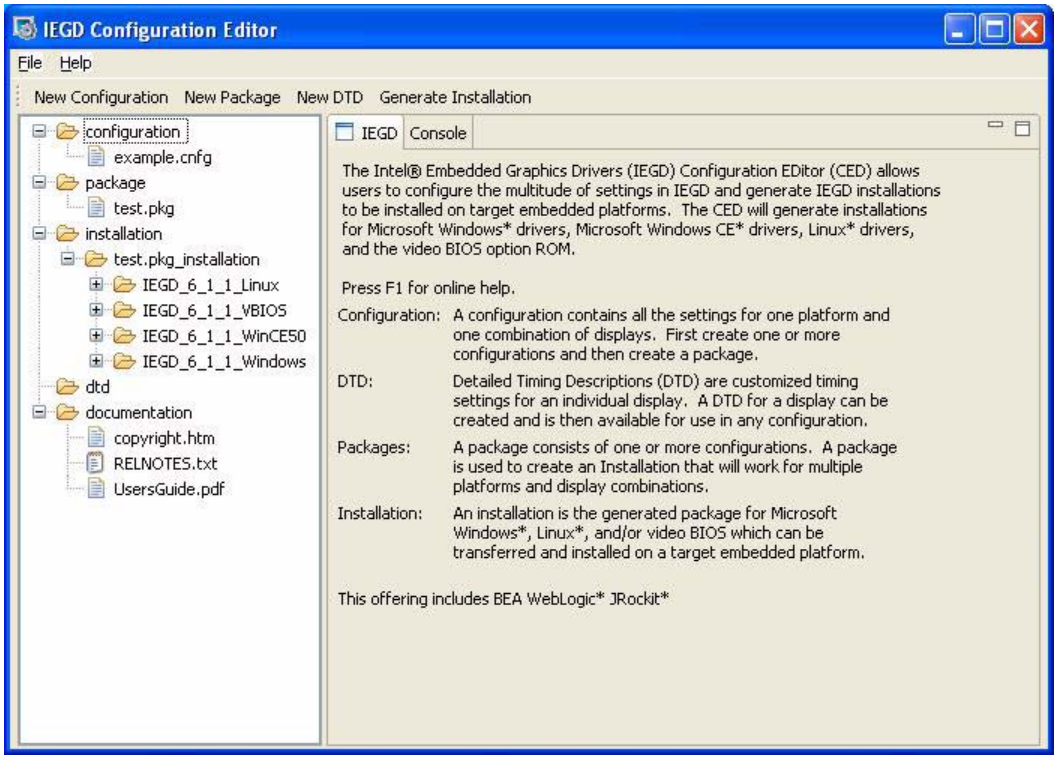
IEGD 8.0 features a Configuration Editor (CED) GUI that facilitates pre-installation configuration of all supported operating systems and Video BIOS. For information on how to use CED, please see the CED online help system.

3.1.1 Creating a Configuration in CED

The following steps present a sample CED configuration.

1. If needed, select **New DTD**.
   - — Choose the DTD Type that most closely aligns with your display parameters, enter parameters, and then click **Finish**.
2. Select **New Configuration**.
   - — Select the chipset, ports, port drivers, DTDs, etc., for the configuration and then click **Finish**.
3. Select **New Package**.
   - — Select the configurations for your package and the platforms for the installation and then click **Finish**.
4. Select the created package then select **Generate Installation**.
   The generated files are placed in the installation folder. The zip files (for Linux, Windows CE, and Windows) contain the generated XFree86, iegd.reg, or INF file.

Figure 4. Sample CED Configuration Start Page



3.2 Configuration Overview

Some aspects of configuring the Intel® Embedded Graphics Drivers are common across the Video BIOS (VBIOS) and the drivers for the supported operating systems. The following sections provide an overview for configuring both the VBIOS and the Intel Embedded Graphics Drivers and describe in detail the common components and tools. This section also describes how to configure the system BIOS for the supported systems.

3.3 System BIOS Settings

Before installing the Intel Embedded Graphics Drivers, you must first configure the system BIOS. The following sections describe the required settings. These descriptions are based on AMIBIOS8* from American Megatrends, Inc., which is the recommended system BIOS to use with the Intel Embedded Graphics Drivers. Settings may vary if a different system BIOS is used.

3.3.1 GMCH PCI Device Enabling

The PCI Device Enabling feature on the Graphics and Memory Controller Hub (GMCH) should be set as specified in Table 8.

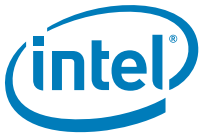


Table 8. GMCH Device 2, Function 1 BIOS Setting

OS	Chipset
	Intel® Q35, Intel® GME965, Intel® Q965, Intel® 945GM, Intel® 945G, Intel® 915GME, Intel® 915GV, Intel® 910GMLE, Intel® 855GME, Intel® 852GME, Intel® 852GM
Microsoft Windows XP* and Microsoft Windows XPe*	Disabled
Microsoft Windows CE*	Disabled
Linux	Disabled

3.3.2 Graphics Mode Select (GMS)

The System BIOS typically allows a portion of physical memory to be dedicated to firmware and graphics driver use. This dedicated memory is known as stolen memory since it is not available to the operating system. The size of this memory is selectable and chipset-specific. Stolen memory is typically used by the firmware and graphics driver to locate the framebuffer, but can also be used as scratch and surface memory. Because it is programmatically set aside during boot by the System BIOS, access to it is direct and does not require OS memory allocation services. Firmware is fully responsible for stolen memory management.

Graphics Mode Select (GMS), or stolen memory, can be set to any of the sizes listed in Table 9. Smaller sizes limit the framebuffer size during firmware boot. Larger sizes marginally increase surface allocation performance for the graphics driver.

Table 9. GMS Settings

Chipset	GMS Settings
Intel® Q35, Q965/ GME965	0, 1 Mbyte, 4 Mbytes, 8 Mbytes, 16 Mbytes, 32 Mbytes, 48 Mbytes, 64 Mbytes
Intel® 945G/ 945GME	0, 1 Mbyte, 8 Mbytes
Intel® 915GV/ 915GME/910GMLE	0, 1 Mbyte, 8 Mbytes
Intel® 852GM/ 852GME/855GME	0, 1 Mbyte, 4 Mbytes, 8 Mbytes, 16 Mbytes, 32 Mbytes

3.3.3 AGP (Accelerated Graphics Port) Aperture Size

The AGP Aperture size can be set to 64 Mbytes or 128 Mbytes. This controls the total amount of graphics memory that can be mapped in the AGP Aperture.

3.4 VBIOS and Driver Configuration

The Intel Embedded Graphics Suite allows user configuration of both the VBIOS and graphics driver as well as programming of Detailed Timing Descriptors (DTDs) for EDID-less panels for both the VBIOS and graphics driver. This is accomplished using CED, which offers several ways to input DTDs, each associated with a potential target panel and display mode for the system. CED generates DTD and configuration settings used by the IEGD VBIOS, Linux, and/or Windows drivers.

Table 10. Parameter Configuration Format (Sheet 1 of 8)

Name	Range / Value	Description
pd	List of port drivers used for the VBIOS. For example: sii164 = 1 ch7009 = 0 ns2501 = 0 etc... (Note that this information pertains to VBIOS — not IEGD — port driver configuration)	This parameter enables port drivers for the VBIOS. You can enable as many port drivers as you want but there is a space limitation of 64K available for port drivers. If you exceed this amount of space, the VBIOS will fail to build. You should only enable the port drivers that are required. 1 to enable a driver. 0 to disable a driver Note: You can select only one from the following list of port drivers: CH7009, CH7017, FS454
configid	Integer (1-15)	Optional keyword used to specify which configuration is used. The config ID specified here must match one of the configuration IDs defined with CED. If this keyword is omitted, all configurations specified in the config file are used. Note that this keyword is not required for Linux and VBIOS configurations.
config	Integer (1-15)	More than one configuration is valid.
comment		A quoted string used to identify the origin of the .bin or .inf file.
name		A quoted string used to identify the configuration name. Name is a required field for VBIOS configuration.
general		Settings that are generic to the configuration.
displayconfig	1 – Single 2 – Clone 4 – Twin 8 – Extended Default: 8	Used to configure initial state of attached displays. 1 – Single. A single display. 2 – Clone. Primary and secondary displays enabled and configured with separate timing pipes. This allows different timings to be applied to each display. Resolutions can be different on both displays. 4 – Twin. Primary and secondary displays are enabled, but with only a single pipe. Both displays share the same resolutions and timings. 8 – Extended. Configures separate pipes to allow primary and secondary displays to have different resolutions and display different content. Upon first boot after the driver installation, this option will only enable the primary display, as the extended modes must be enabled in the operating system (i.e., **Extended Desktop** in the **Display Properties** sheet within Microsoft Windows).
displaydetect	0 - Disable 1 - Enable	Enable or disable Display Detection. Note that this parameter must be Enabled in order to use `COMMON_TO_PORT` values. Default is 0. Please see Section 3.5, "Display Detection and Initialization" on page 35 for detailed information on this parameter.

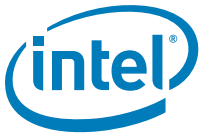


Table 10. Parameter Configuration Format (Sheet 2 of 8)

Name	Range/Value	Description
PortOrder	PortOrder must be specified as a quoted string containing five digits. The valid values are: 1 - Integrated TV Encoder (mobile chipsets only) 2 - DVO/sDVO B port 3 - DVO/sDVO C port 4 - Integrated LVDS port (mobile chipsets only) 5 - Analog CRT port Default: 0 for all keys	Search order for detecting attached displays for the Display Detection feature. When Display Detection is enabled, the PortOrder determines which display is primary and which display is secondary. The port search order can be specified to ensure the port device (DVO device) is found, based on the system integrator's routing choices. Default ordering is chosen by specifying zeros in the PortOrder keys. Default ordering is chipset specific; see Table 46, "Default Search Order" on page 176. Please see Section 3.5, "Display Detection and Initialization" on page 35 for more information on using `PortOrder` in combination with the Display Detect feature.
clonewidth cloneheight	Typical sizes: clonewidth – 800, cloneheight - 600 clonewidth – 1024, cloneheight - 768 clonewidth – 1280, cloneheight - 768 clonewidth – 1400, cloneheight – 1050	Width and height for a cloned display.
clonerefresh = 60	Typical refresh rates (expressed in Hz): 60 Hz, 75 Hz, 85 Hz	Refresh rate for a cloned display.
OverlayOff	0 - Overlay on (default) 1 - Overlay off	This parameter allows you disable Overlay support, which is enabled by default. **Note:** This parameter is only for Microsoft Windows* and Microsoft Windows CE. The Linux* configuration for the XF86Config provides a standard option that performs the same function.
No_DFB	0 - Off (Default) 1 - On	This parameter enables the IEGD to pass the DIB call back to the OS. This is required in certain circumstances to improve performance.
vbios		This block contains settings for the new Video BIOS. Note that you only need to specify the parameters you are actually using. You do not need to specify all the parameters in this block. If you omit any parameters, the vbios uses the default values.
COMMON_TO_PORT	6 digit value	Maps the ports from the system BIOS to a port number used by the graphics hardware. Please see Section 4.3.2, "Configuring the Video BIOS" on page 53 for more information on this parameter. Note that the `displaydetect` parameter must be set to Enabled in order for the `COMMON_TO_PORT` values to be used. The default is all zeroes: 000000

Table 10. Parameter Configuration Format (Sheet 3 of 8)

Name	Range / Value	Description
post_display_msg	0 - disable greater than 0 - enable and display POST message for the specified number of seconds	Enables or disables the POST (Power On Self Test) message. When you specify a value greater than 0, the message is displayed for the specified number of seconds. For example: post_display_msg = 5 This enables the POST message and displays it for approximately 5 seconds. The maximum value that can be entered here is 65535. The default is 1, enable and display the POST message for approximately 1 second.
oem_string	double-quoted string	This string appears on the display when the post_display_msg is enabled and the VBIOS starts up. The maximum string length is 100 characters. The default is " " (two double quotes with a single space in between).
oem_vendor	double-quoted string	This string appears on the display when the post_display_msg is enabled and the VBIOS starts up. The maximum string length is 80 characters. The default is " " (two double quotes with a single space in between).
oem_product_name	double-quoted string	This string appears on the display when the post_display_msg is enabled and the VBIOS starts up. The maximum string length is 80 characters. The default is " " (two double quotes with a single space in between).
oem_product_rev	double-quoted string	This string appears on the display when the post_display_msg is enabled and the VBIOS starts up. The maximum string length is 80 characters. The default is " " (two double quotes with a single space in between).

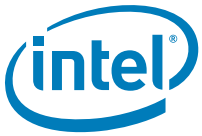


Table 10. Parameter Configuration Format (Sheet 4 of 8)

Name	Range / Value	Description
int15	5 digits	This parameter allows you to enable or disable the five System BIOS 15h interrupt hooks. The value must be 5 digits in length. Each digit is associated with one of the five System BIOS interrupt 15h hooks as shown below (left to right) 1 - 5F31h, POST Completion Notification Hook 2 - 5F33h, Hook After Mode Set 3 - 5F35h, Boot Display Device Hook 4 - 5F36h, Boot TV Format Hook 5 - 5F38h, Hook Before Set Mode (Please see Appendix C for more information on 5F functions.) The value of each digit must be a 0 or a 1 as follows: 0 - disable a System BIOS 15h hook 1 - enable a System BIOS 15h hook For example, `int15 = 11001` Enables 5F31h, 5F33h, and 5F38h hooks only. The 5F35h and 5F36h hooks are disabled. The default is 11111, enable all five hooks.
default_refresh_0 default_refresh_1	Valid refresh rate in Hz. (specified in decimal, e.g., `default_refresh_0=60`).	These settings allow you to specify which refresh rate is used for certain VESA modes on the primary and secondary displays. For example, mode 0x117 specifies refresh rates of 60 Hz, 75 Hz, and 85 Hz. This setting allows you to specify which of those three rates to use. The default is 60 for both parameters.
default_mode_0 default_mode_1	Valid VGA or VESA mode. (e.g., `default_mode_0 = 3`)	These settings establish the default VGA or VESA mode to use for the primary (0) and secondary (1) displays. The values should be set to a valid standard VGA or VESA mode (in hexadecimal format, for example, 0x117). Note that if you select a VGA mode, the secondary display will automatically be set to this mode since the VBIOS can only support one simultaneous VGA mode. For a list of valid VGA and VESA modes, refer to Table 15 on page 58 and Table 16 on page 59. The default is mode 3 for both.
default_vga_height	Valid VGA height	This setting allows you to specify which resolution is used for certain VGA modes. Since only one VGA mode can be supported on both displays, this setting applies to the primary display mode (`default_mode_0`) only. For example, mode 3 specifies three possible resolutions: 640x200, 640x350, and 720x400. In this example, setting `default_vga_height=350` indicates the resolution 640x350.

Table 10. Parameter Configuration Format (Sheet 5 of 8)

Name	Range / Value	Description
port	1 - Integrated TV Encoder (mobile chipsets only) 2 - DVO/sDVO B port 3 - DVO/sDVO C port 4 - Integrated LVDS port (mobile chipsets only) 5 - Analog CRT port	Used to define port specific settings.
name		A quoted string used to identify the port.
general		Settings that are generic to the port.
rotation	Windows* Range: 0x0 or 0 – 0 degrees 0x5A or 90 – 90 degrees 0xB4 or 180 – 180 degrees 0x10E or 270 – 270 degrees Linux Range: 0 – 0 degrees 90 – 90 degrees 180 – 180 degrees 270 – 270 degrees Default: 0	Rotation of the display.
flip	Windows: 0x0 or 0 – turn off horizontal flip 0x1 or 1 – turn on horizontal flip Default: 0 Linux Boolean: on - horizontal flip off - no horizontal flip Default: off	Flip of the display.
centeroff	Default: 0 – disabled, allow centering and add compatibility modes 1 – enabled, no centering, no added compatibility modes	When this option is enabled it DISABLES centering. Also, depending on the combination of "edid" + "user-dtd" + connected hardware, IEGD will add missing compatibility modes (6x4, 8x6, 10x7& 12x10) via centering. Use this option to disable this feature.
edid	0 – Do not read EDID from panel/CRT 1 – Attempt to extract EDID timing data from panel/CRT	If VBIOS/Driver reads EDID from panel/CRT.

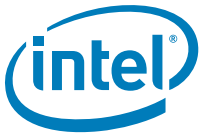


Table 10. Parameter Configuration Format (Sheet 6 of 8)

Name	Range/Value	Description
edid_avail edid_not_avail	Range [16 bits] Valid values (specified in hex): bit 0 ---------- 0 - Do not use driver built-in standard timings 1 - Use driver built-in standard timings bit 1 (not applicable to edid_not_avail) ---------- 0 - Do not use EDID block 1 - Use EDID block and filter modes bit 2 ---------- 0 - Do not use user-defined DTDs 1 - Use user-defined DTDs bit3 - bit15 ---------- Reserved for future use.	These two parameters are used to control the available timings for any display. edid_avail is used when EDID values are read from the display. If an attempt to read EDID from the display fails or the edid parameter is set to 0, then the driver uses the edid_not_avail flags. The value for both parameters must be specified as a hex value. Defaults: edid_avail: 3 (hex). Bit 0 = 1, Bit 1 = 1, Bit 2 =0 (Use driver built-in standard timings and EDID block and filter modes.) edid_not_avail: 1 (hex). Bit 0 = 1, Bit 1 = 0, Bit 2 = 0. (Use driver-built-in standard timings.) Please see Section 3.6, "Advanced EDID Configuration" on page 38 for detailed information.
multidvo	0 – Do not attempt to detect a second decoder of same type 1 – After detect of a decoder, continue to attempt detection of same type of decoder until fail	If VBIOS/Driver detects a second decoder of same type. This value is hard-coded to "1" for Windows configuration and will ignore this setting.
dvo		DVO device information.
i2cpin	<0-6>	The GPIO pin pair used on the I^2C bus to read and write to DVO device registers. For pin pair definitions, see Table 11, "I2C/DDC Pin Pair Definitions for 8x Chipsets" on page 34.
ddcpin	<0-6>	The GPIO pin pair used as DDC bus to read panel EDID data. For pin pair definition, see Table 11, "I2C/DDC Pin Pair Definitions for 8x Chipsets" on page 34.
i2cdab	<0x00-0xff>	I^2C device address for reading and writing device registers. The device address should be in 8-bit format with the 7-bit slave address assigned to its bits 7:1 and bit 0 set to 0.
ddcdab	<0x00-0xff>	I^2C device address for reading EDID data from display through the DDC bus.
i2cspeed	[10-400]. Units in KHz	Speed of I^2C bus for DVO device.
ddcspeed	[10-400]. Units in KHz	Speed of I^2C bus for EDID device.
fpinfo		Panel-specific information.
bkltmethod	Range [0-3] 0 – no backlight 1 – Port Driver 2 – GMCH 3 – ICH	Instructs which backlight method is required for the panel attached to the given port. If zero is supplied, or the key is not present, then no backlight control is provided.

Table 10. Parameter Configuration Format (Sheet 7 of 8)

Name	Range / Value	Description
bkltt1	Range [0 -0xfff]. Units in ms	(T1) Time delay between VDD active, and DVO clock/data active. Zero indicates no delay required.
bkltt2		(T2) Time delay between DVO clock/data active and Backlight enable.
bkltt3		(T3) Time delay between Backlight disable and DVO clock/data inactive.
bkltt4		(T4) Time delay between DVO clock/data inactive and VDD inactive.
bkltt5		(T5) Minimum delay between VDD inactive, and active.
gpiopinvee	Valid ICH GPIO pin, 0 indexed For example: gpiopinvdd = 3 gpiopinvee = 5 gpiopinenable = 1	GPIO connection for panel power.
gpiopinvdd		GPIO connection for backlight power on/off sequencing signal.
gpiopinbklt		GPIO to enable backlight signal.
UseGMCHClockPin	1 - Flat panel is connected to the clock pin 0 - Flat panel is not connected to the clock pin	This entry is needed when GMCH is selected as backlight control method.
UseGMCHDataPin	1 - Flat panel is connected to the data pin 0 - Flat panel is not connected to the data pin	This entry is needed when GMCH is selected as backlight control method.
dtd		Denotes a Detailed Timing Descriptor (DTD) block. Settings in this section, except for the `flags` parameter, correspond to the Detailed Timing Block described in the VESA standard "Extended Display Identification Data Standard", Version 3, November 13, 1997.
p_clock	Range [0-0x7fffffff]	Pixel clock value in KHz.
h_active	Range 0-4096 [12 bits]	Horizontal Active.
v_active	Range 0-4096 [12 bits]	Vertical Active.
h_sync	Range 0-1024 [10 bits]	Horizontal Sync Offset.
v_sync	Range 0-64 [6 bits]	Vertical Sync Offset.
h_syncp	Range 0-1024 [10 bits]	Horizontal Sync Pulse Offset.
v_syncp	Range 0-64 [6 bits]	Vertical Sync Pulse Width.
h_blank	Range 0-4096 [12 bits]	Horizontal Blanking.
v_blank	Range 0-4096 [12 bits]	Vertical Blanking.
h_border	Range 0-256 [8 bits]	Horizontal Border. Currently not supported.
v_border	Range 0-256 [8 bits]	Vertical Border. Currently not supported.
h_size	Range 0-4096 [12 bits]	Horizontal Size. Currently not supported.
v_size	Range0-4096 [12 bits]	Vertical size. Currently not supported.

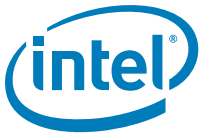


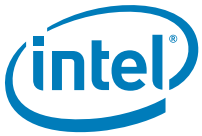
Table 10. Parameter Configuration Format (Sheet 8 of 8)

Name	Range / Value	Description
flags	Range [32 bits] Valid values: bit 31 ---------- 0 - Non-interlaced 1 - Interlaced bit 27 ---------- 0 - vertical sync polarity active low 1 - vertical sync polarity active high bit 26 ---------- 0 - horizontal sync polarity active low 1 - horizontal sync polarity active high bit 25 ---------- 0 - blank sync polarity active high 1 - blank sync polarity active low bit 17 ---------- 0 - Normal DTD 1 - Panel/display Native DTD All other bits ---------- Do not use any other bits; all other bits must be set to 0.	Interlace, Horizontal polarity, Vertical polarity, Sync Configuration, etc. Note that these flags are IEGD specific and do not correspond to VESA 3.0 flags. For example, to set Interlaced with Horizontal Sync Polarity high (bits 31 and 26), then the flags value = 0x84000000. (Binary = 10000100000000000000000000000000)
attr	0-0xFFFF	Attribute values that are specific to the DVO device for the port. See Appendix B, "Port Driver Attributes" for specific attribute IDs and associated values.
id <Attribute ID>	0 - 2^{32}	id = <value>. Both the Attribute ID and its value should be specified in decimal. For example, to set brightness to 50, you specify id 0 = 50 See Appendix B, "Port Driver Attributes".

Table 11. I²C/DDC Pin Pair Definitions for 8x Chipsets

Pair #	Signal Name	Signal Description	Notes	Intel 852GME, 855GME Chipset Pin Names
0	DDCA	CRT DDC for Analog monitor (CRT) connection.	This cannot be shared with other DDC or I²C pairs due to legacy monitor issues.	DDCACLK DDCADATA
1	LCLKCTL	SCC Chip - For control of SSC clock generator devices down on motherboard.	If SSC is not supported then can be used for DVOB or DVOC GMBUS.	LCLKCTLA LCLKCTRLB
2	DDCP	Panel DDC for Digital Display connection via the integrated LVDS display port for support for EDID panel.	If EDID panels are not supported. Can optionally use as GMBUS for DVOB or DVOC.	DDCPCLK DDCPDATA
3	MDVI	DVI 1 DDC - GMBUS control of DVI devices (TMDS or TV encoder)	Can optionally use as GMBUS for DVOB or DVOC.	MDVICLK MDVIDATA
4	MI2C	DVO I²C - GMBUS control of DVI devices (TMDS or TV encoder)	Can optionally use as GMBUS for DVOB or DVOC.	MI2CCLK MI2CDATA
5	MDDC	DVI 2 DDC - DDC for Digital Display connection via TMDS device	Can optionally use as GMBUS for DVOB or DVOC.	MDDCCLK MDDCDATA
6	EXTTS	External Thermal Sensor Input		EXTTS_0

Note: Do not attempt to change pin pair settings for the Mobile Intel® 910GMLE Express, Intel® 915GV Express, Mobile Intel® 915GME Express, Intel® 945G Express, Mobile Intel® 945GM Express, Intel® Q965 Express, Mobile Intel® GME965 Express, and Intel® Q35 Express chipsets.



3.5 Display Detection and Initialization

The Display Detection and Initialization feature, when enabled, automatically detects displays and allocates ports without the need to change any configuration files. This feature is off by default and can be enabled either through CED or by directly editing the `iegd.inf` file for Microsoft Windows or the `XF86Config/xorg.conf` file for Linux.

To enable the feature via CED, select the `DisplayDetect` option on the CED Chipset Configuration page. Please see Section 3.1, "IEGD Configuration Editor (CED)" on page 23 or CED online help for more information.

Alternatively, you can enable the feature in Microsoft Windows by entering the following line in the `[iegd_SoftwareDeviceSettings]` section of the `iegd.inf` file:

HKR, All\<ConfigID>\General, DisplayDetect, %REG_DWORD%, 1

where <ConfigID> is the configuration ID (without the angle brackets).

To enable the feature in Linux, enter the following line Option setting in the `XF86.conf` or `xorg.conf` file:

Option "Config/<ConfigID>/General/DisplayDetect" "1"

When the display detection feature is enabled, ports are allocated only when the display satisfies the following conditions:

1. The port is not in use (that is, it is not already allocated).
2. The display is detected by the port driver.

The first port that passes these conditions is allocated. If condition 2 fails for all ports, then the first port in the `PortOrder` setting that passes condition 1 is allocated. If the port is not detectable (specifically the internal LVDS or external LVDS using CH7017), the driver assumes the display is connected. Condition number 2. always passes for these displays.

When this feature is disabled, display allocation is done based on `PortOrder` and no display detection is performed.

3.5.1 Display Detect Operation

This section describes the logic of the Display Detection feature and provides several examples.

1. If Display Detect is disabled, the driver uses the first two ports identified in the PortOrder.
2. If Display Detect is enabled and you are using the 8.0 version of the VBIOS, the VBIOS performs the display detection. The driver then checks to see if the VBIOS returns the display allocations and if it does, the driver does not re-execute the display detection steps.

   If you are using the Legacy VBIOS, then the driver performs display discovery as described in the following steps.
3. The number of displays to be detected is based on the `DisplayConfig` settings in the configuration. If this is set to `Single`, then only one display is detected. If it is set to any other value, a maximum of two displays will be detected.

4. The IEGD goes through each port in the `PortOrder` settings and attempts to detect a display using the following algorithm:

   a. If a display is detected, it is based on the PortOrder sequence. Display allocation of the port is performed once the display has been detected. For example:

      PortOrder = "53240" (CRT, DVOC, DVOB, LVDS)
      Displays Connected = DVOB, CRT

      Primary display allocation: Searches for a display connected according to the PortOrder sequence. The first detected display is a CRT, so the Primary display is CRT.

      Secondary display allocation: Searches for a display connected according to the PortOrder sequence. The first detected and non-allocated display is DVOB, so the Secondary display is DVOB.

   b. If no display is detected on any of the ports, then the DisplayDetect option is turned off and ports are allocated in the order defined by PortOrder. For example:

      PortOrder = “32000” (DVOC, DVOB)
      Displays Connected = None

      Primary display allocation: Searches for a connected display according to the PortOrder. Because no displays are detected, the Primary display is set to DVOC.

   c. The driver cannot detect the presence of a display connected to the Internal LVDS and external LVDS displays connected to some DVO devices (for example, an LVDS connected to the CH7017). Consequently, the driver assumes that an LVDS display is connected if it is in the PortOrder. If you only want to use the internal LVDS when no CRT and DVO devices are connected, then put LVDS in the PortOrder after them. For example:

      PortOrder = “53240” (CRT, DVOC, DVOB, LVDS)
      Display Connected = None

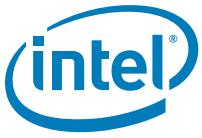
      Primary display allocation: Searches for a display connected according to PortOrder sequence. Since no display is connected and since LVDS is specified in the PortOrder, the driver assumes that an LVDS display is connected. Consequently, the Primary display is set to LVDS.

   d. Since the driver cannot detect the presence of a display connected to the Internal LVDS and certain external LVDS displays, it therefore always assumes that they are connected if they are listed in the PortOrder. Be careful not to set the PortOrder that prevents the driver from detecting a connected display. For example:

      PortOrder = “54320” (CRT, LVDS DVOC, DVOB)
      Displays Connected = CRT, DVOC

      Primary display allocation: Searches for a connected display according to the PortOrder. In this case, the Primary display is set to CRT.

      Secondary display allocation: Searches for a connected display according to the PortOrder. Even though DVOC is connected, the driver assumes that the internal LVDS is also connected. Consequently, the driver never detects the display connected to the DVOC port. To change this, move DVOC before LVDS in the PortOrder (“53420” rather than “54320”).



e. If the port drivers are not loaded for any of the ports specified in the PortOrder, the driver enables port 5 (CRT) only. For example:

PortOrder = “32000” (DVOC, DVOB)
PortDrivers = “” (None)

Primary display allocation: Searches for displays connected according to the PortOrder. Since no port drivers are available for the specified ports, CRT port 5 is enabled. Consequently, the Primary display is set to CRT.

3.5.2 Detectable Displays

Table 12 provides a list of displays that are detectable by the IEGD.

Table 12. Detectable Displays

Transmitter	Display Type	Detectable by IEGD?
GMCH Analog CRT	VGA	Yes
GMCH Integrated LVDS	LVDS	No (assumed attached)
GMCH Integrated TV Out	TV Out	N/A
CH7009	DVI	Yes
CH7009	TV Out	Yes
CH7017	LVDS	No (assumed attached)
CH7017	TV Out	Yes
CH7021	VGA Bypass	TBD
CH7307	DVI	Yes
CH7308	LVDS	No (assumed attached)
CH7315	HDMI/DVI	TBD
CH7317	VGA Bypass	TBD
CH7319	DVI	Yes
CH7320	DVI	Yes
Sil 164	DVI	Yes
Sil 1362	DVI	Yes
Sil 1364	DVI	Yes
TFP410	DVI	Yes
TH164	DVI	Yes
NS2501	LVDS	Yes
NS387R	LVDS	Yes
FS454	TV Out	Yes

3.6 Advanced EDID Configuration

Shown in the following EDID Options example, the If EDID Device (edid_avail) and If Not EDID Device (edid_not_avail) options in CED are found on the CRT, DVO, LVDS, and TV Out configuration pages.

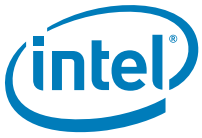
These options control the available timings for any display. The edid_avail parameter is used when EDID information is read from the display. If the driver is unable to read EDID information from the display or if the edid parameter is set to "0" (disable), then the settings of the edid_not_avail parameter are used.

The default behavior of edid_avail is to use the driver's built-in standard timings and EDID block and filter modes. The default for edid_not_avail is to use the driver's built-in standard timings. Please see Table 10 for more information on these parameters.

The IEGD supports three different types of EDID display modes:

1. **Built-in display modes**. These modes are hard-coded in the IEGD. These modes can be filtered based on the EDID block.
2. **EDID-DTDs**: These are Detailed Timing Descriptors read from the EDID block. EDID can have these DTDs along with other information about the display.
3. **User-specified DTDs** defined in CED. See Section 3.6.2.

The Advanced EDID Configuration supports different possible combinations of display modes when an EDID display is present along with user-specified DTDs.



3.6.1 Sample Advanced EDID Configurations

Table 13 presents various EDID configurations and the EDID settings in CED used for those configurations.

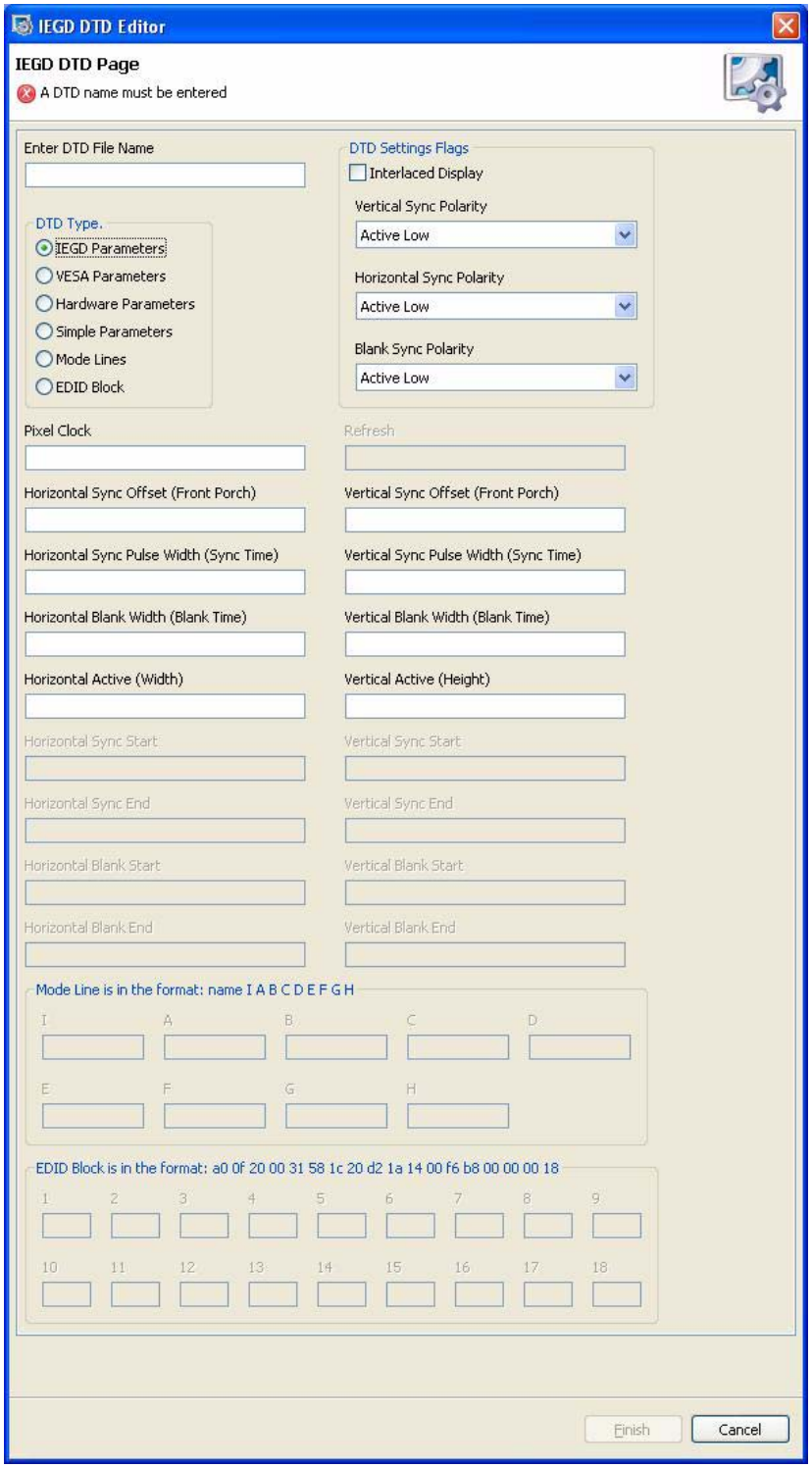
Table 13. Sample Advanced EDID Configurations

Configurations	CED Settings	Description
1. Use only filtered built-in and any EDID-DTDs when the display has EDID information 2. Use all built-in modes when the display doesn't contain EDID information	edid = 1 edid_avail = 3 edid_not_avail = 1	Default values.
1. Use only filtered built-in modes and EDID-DTDs when the display has EDID. 2. Use only user-DTDs otherwise.	edid = 1 edid_avail = 3 edid_not_avail = 4	This configuration allows the IEGD to use its built-in display modes and the modes provided by the display. If the IEGD is unable to read EDID information from the display, then the IEGD uses the user-DTDs defined in CED.
1. Use only user-DTDs regardless of connected display. (Typically used for a custom panel that only supports user-defined DTDs.) 2. Use limited set of timings when a panel EDID is present, but the IEGD cannot read the EDID information.	edid = 0 edid_avail = (any value) edid_not_avail = 4	Only user-DTDs defined in CED are used.
1. Use EDID-DTDs for an EDID display. 2. Use user-DTDs for a non-EDID display.	edid = 1 edid_avail = 2 edid_not_avail = 4	This configuration uses the EDID-DTDs when an EDID display is detected and EDID information is read from the display. If the driver detects a non-EDID display, then the IEGD uses user-DTDs defined in CED.
1. Use only EDID-DTDs and user-DTDs for an EDID display. 2. Use user-DTDs only for a non-EDID display.	edid = 1 edid_avail = 6 edid_not_avail = 4	This configuration uses both EDID-DTDs and user-DTDs when the IEGD detects an EDID display. If the driver detects a non-EDID display, then the IEGD uses user-DTDs defined in CED.

3.6.2 User-Specified DTDs

CED provides the ability to input DTD data directly. There are numerous sources of DTD data: VESA, panel manufacturers, etc. Figure 5 shows an example DTD Editor page, and Table 14 presents DTD parameter descriptions.

Figure 5. IEGD DTD Editor



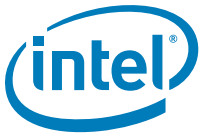


Table 14. DTD Parameter Descriptions

Parameter	Description
Enter DTD File Name	Enter a name for this customized DTD. This is a required field and the name must be between 1 and 50 characters and may contain spaces and underscores.
DTD Type	Select the DTD Type that most closely aligns with your display parameters. Options are: • IEGD Parameters • VESA Parameters • Hardware Parameters • Simple Parameters • Mode Line • EDID Block
IEGD Parameters	The IEGD Parameters are the same as the current PCF/CED DTD parameters.
VESA Parameters	The VESA Parameters allow the user to create a DTD from a VESA monitor timing standard.
Hardware Parameters	The Hardware Parameters are the parameters that are used by IEGD.
Simple Parameters	The Simple Parameters (CVT Standard) is a process for computing standard timing specifications. The method for developing Reduced Blanking timings is not included.
Mode Line	The Mode Line is a video timing spec used by XFree86. The Xfree86 timing setting for Mode Line is “name” I A B C D E F G H. For example: “640x480@8bpp” 25.175 640 672 728 816 480 489 501 526.
EDID Block	The EDID Block is the detailed timing section (18 bytes) of the basic 128-byte EDID data structure. The detailed timing section starts at 36h of the 128-byte EDID data structure. Enter the EDID block 1 byte at a time. Example: a0 0f 20 00 31 58 1c 20 d2 1a 14 00 f6 b8 00 00 00 18
Horizontal Sync Offset	Specifies the amount of time after a line of the active video ends and the horizontal sync pulse starts (Horizontal Front Porch). Range 0-1023 [10 bits].
Horizontal Sync Pulse Width	Width of the Horizontal Sync Pulse (Sync Time) which synchronizes the display and returns the beam to the left side of the display. Range 0-1023 [10 bits].
Horizontal Sync Start	This parameter specifies the start of the horizontal active time. Range 0-40957.
Horizontal Sync End	This parameter specifies the end of the horizontal active time. Range 0-49148.
Horizontal Blank Width	This parameter indicates the amount of time it takes to move the beam from the right side of the display to the left side of the display (Blank Time). During this time, the beam is shut off, or blanked. Range 0-4095 [12 bits].
Horizontal Blank Start	This parameter specifies the start of one line of the video and margin period. Range 0-32766.
Horizontal Blank End	This parameter specifies the end of one line of the video and margin period. Range 0-65533.
Horizontal Active	Number of pixels displayed on a horizontal line (Width). Range 0-4095 [12 bits].
Pixel Clock	Pixel clock value in KHz. Range 0-0x7fffffff
Refresh	Also known as the Vertical Refresh, the rate the full display updates. Standard refresh rates are 50Hz, 60Hz, 75Hz, and 85Hz.
Vertical Sync Offset	Specifies the amount of time after last active line of video ends and vertical sync pulse starts (Vertical Front Porch). Range 0-4095 [12 bits].

Table 14. DTD Parameter Descriptions (Continued)

Parameter	Description
Vertical Sync Pulse Width	Specifies the Width of the Vertical Sync Pulse which synchronizes the display on the vertical axis and returns the beam to the top, left side of the display. Range 0-63 [6 bits].
Vertical Sync Start	This parameter specifies the start of the vertical sync. Range 0-4157.
Vertical Sync End	This parameter specifies the end of the vertical sync. Range 0-4220.
Vertical Blank Width	The amount of time for the complete vertical blanking operation to complete. It indicates the time it takes to move the beam from the bottom right to the top, left side of the display (Blank Time). During this time, the beam is shut off, or blanked. Range 0-4095 [12 bits].
Vertical Blank Start	This parameter specifies the start of display vertical blanking including margin period. Range 0-4094.
Vertical Blank End	This parameter specifies the end of vertical blanking. Range 0-8189.
Vertical Active	The number of active lines displayed (Height). Range 0-4095 [12 bits].
DTD Settings Flags	This section allows you to set flags for Interlace, Vertical Sync Polarity, Horizontal Sync Polarity, and Blank Sync Polarity. Each field in this section is described below. Interlaced Display: Check for Interlaced Unchecked for Non-interlaced Vertical Sync Polarity: Active Low (Default) Active High Horizontal Sync Polarity: Active Low (Default) Active High ***Note:*** These flags are IEGD-specific and do not correspond to VESA 3.0 flags.

3.7 External PCI Graphics Adaptor as Primary Device

The IEGD can be configured to work with an external PCI graphics adaptor card as the primary graphics adaptor device with the Intel internal graphics device (GMCH) as the secondary graphics device. You can configure your system to boot with a PCI graphics adaptor in the System BIOS. When an external PCI graphics adaptor is designated as the primary graphics adaptor, the Intel GMCH becomes the secondary graphics device.

Note: The term *secondary* adaptor refers to the adaptor that is not the *boot-up*, or VGA-Compatible, adaptor. The secondary adaptor is not necessarily the secondary display as assigned by the OS.

You can configure an external PCI card to work with the IEGD as follows:

- The external PCI card as the primary graphics adaptor and the GMCH internal graphics device as the secondary.
- The external PCI card as the secondary graphics adaptor and the GMCH internal graphics device as the primary.

Note: This feature is not supported on Microsoft Windows CE systems.

The IEGD allows you to specify which display is primary, secondary, and tertiary. It allows Twin and Clone configurations on the internal graphics device when the external PCI display is the primary graphics adaptor. It also allows Twin and Clone configurations on the internal graphics device when the external PCI device is the secondary graphics adaptor.

An external PCI graphics driver runs independently without sharing resources with the IEGD.

The following figures show several configurations when an external PCI adaptor is the primary graphics device and when it is the secondary graphics device.

Figure 6 shows an External PCI card as the primary graphics adaptor card and the IEGD driver as the secondary. The drivers do not share hardware resources. The OS decides the framebuffer content and handles that by drawing to the respective driver independently.

Figure 6. External PCI Graphics Card as Primary Driver and IEGD as Secondary Driver

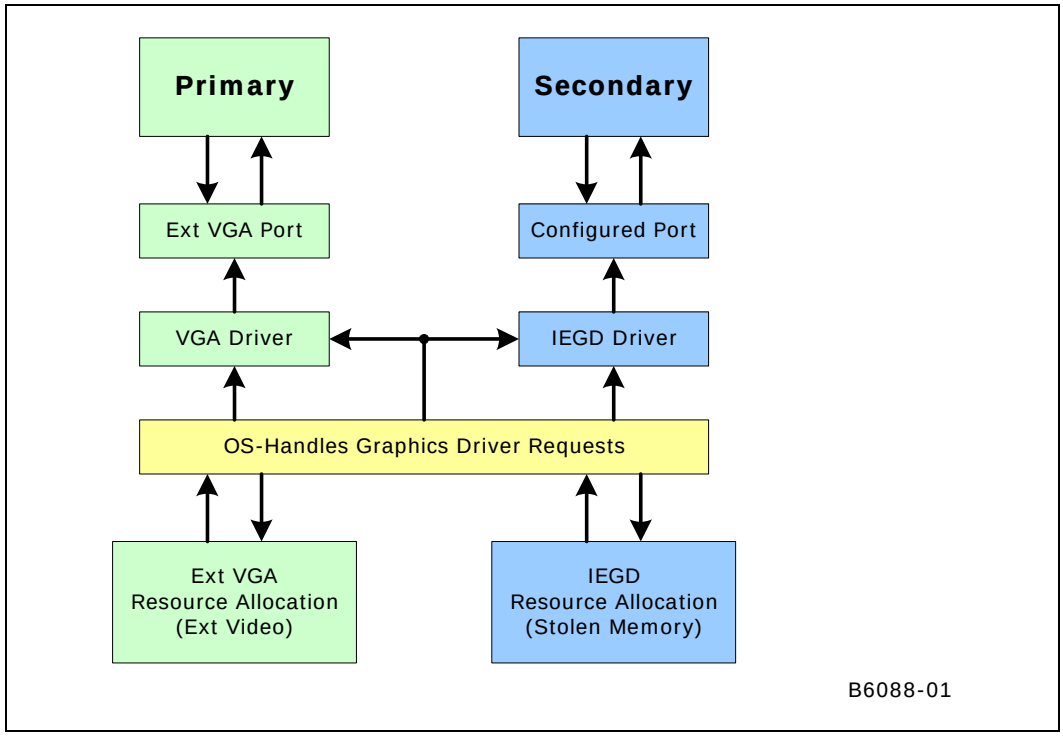


Figure 7 shows the interaction between the IEGD driver and the External VGA driver when the IEGD is booted as the primary driver. Again, the drivers do not share hardware resources. The OS decides the framebuffer content and handles it by drawing to the respective driver independently.

Figure 7. IEGD as Primary Driver and External PCI Graphics Card as Secondary Driver

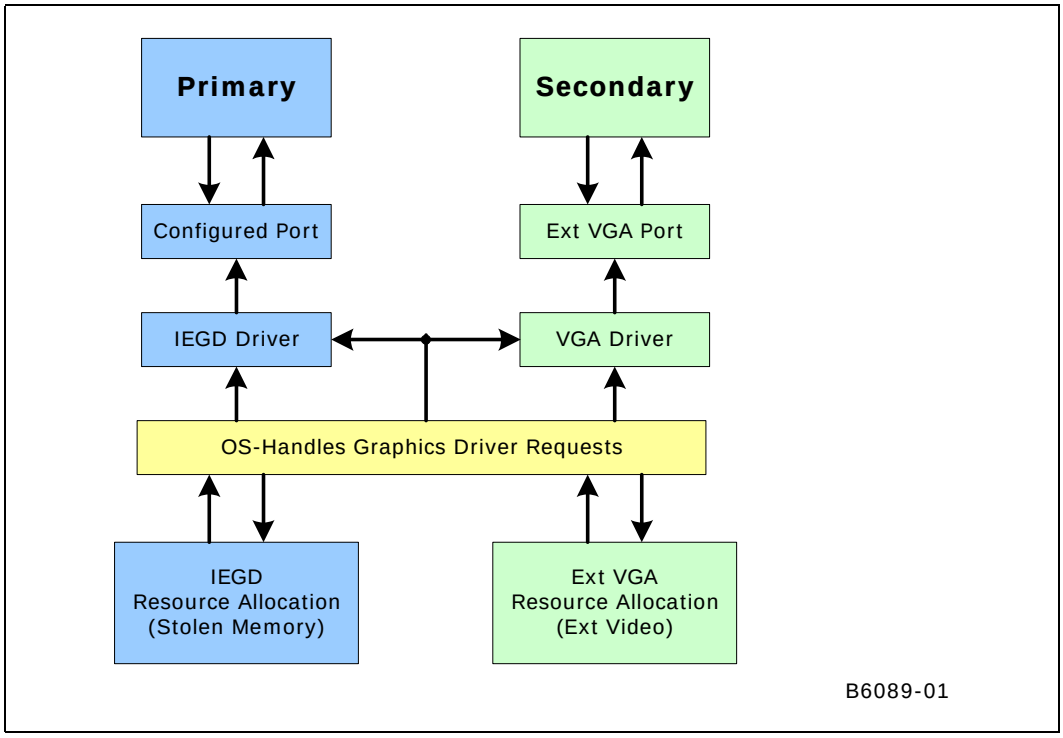
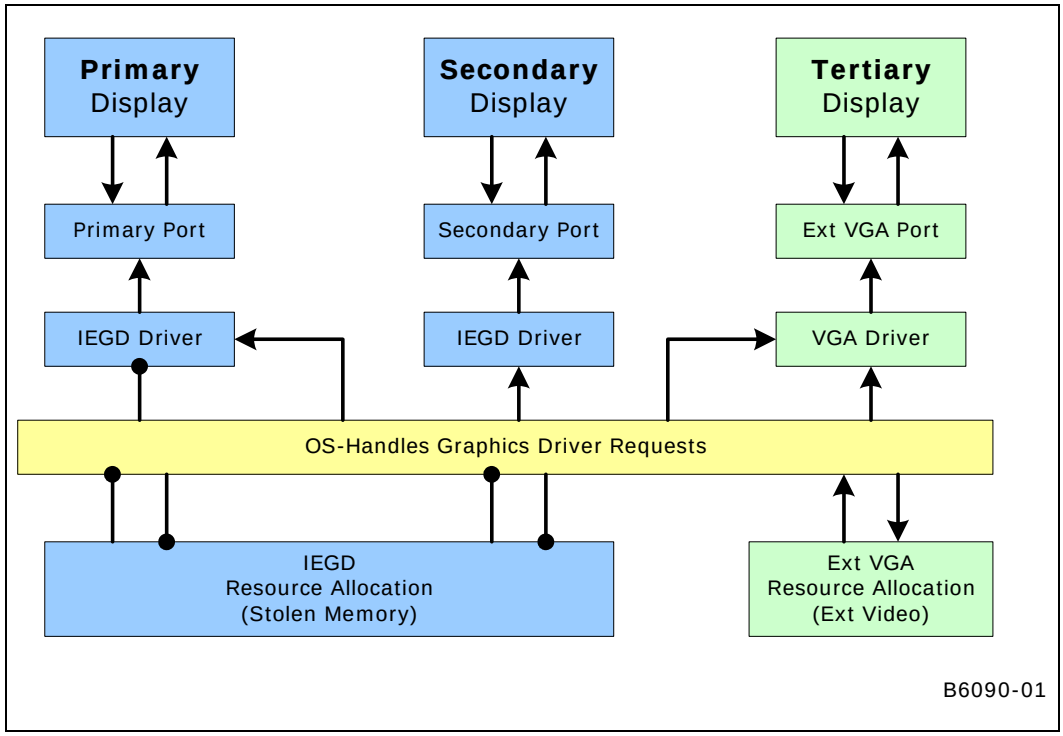


Figure 8 shows a sample configuration where the internal graphics device is primary and configured to use two ports to drive two displays while an external PCI graphics adaptor is used to drive a tertiary display. Note that regardless of the number of ports being assigned to a driver, the external PCI graphics run independently without sharing resources with the IEGD driver.

Figure 8. IEGD as Primary Driver With Two Displays and External PCI Driving a Tertiary Display



3.8 Enhanced Clone Mode Support

The Enhanced Clone Mode feature allows you to specify a clone display size that is different from the primary display. It also allows you to change the clone display size at runtime using the IEGD Runtime GUI (see Section 5.6, “Viewing and Changing the Driver Configuration from Microsoft Windows” on page 72 or Section 7.7, “Runtime Operation” on page 129 for Linux systems.

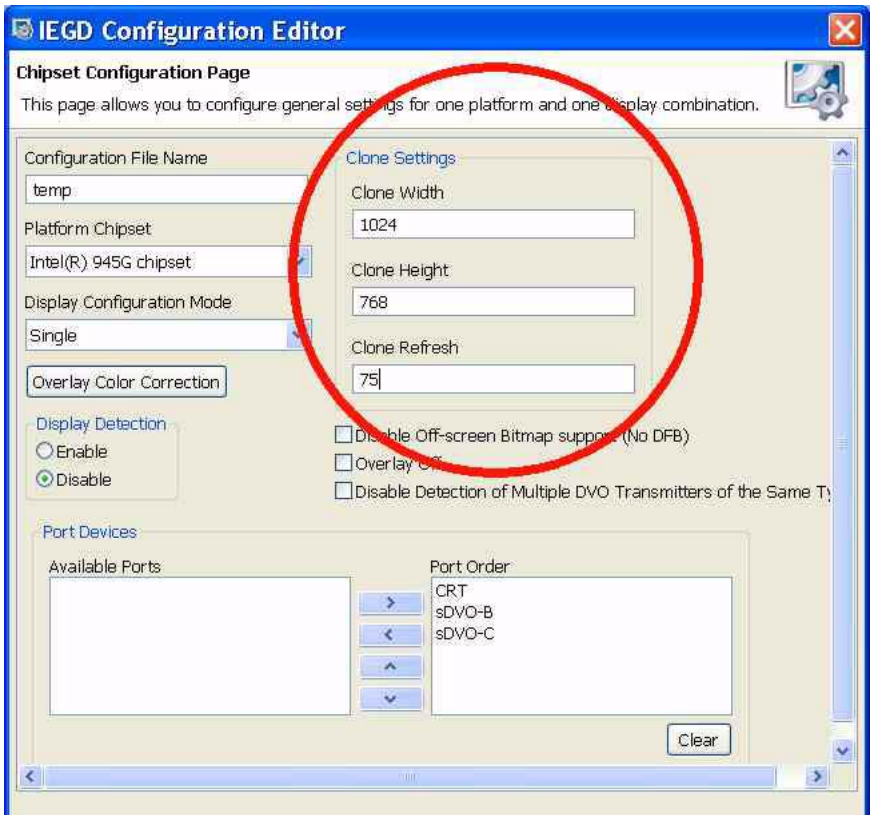
In Clone mode, the framebuffer is always allocated to match primary display size. On the clone display (secondary display) the image is centered if the display is bigger than the framebuffer. Centering is done only if the requested resolution and refresh rate is not available for the clone display.

Extended Clone mode is implemented through the use of three CED parameters:

- clonewidth — allows you to specify a width for the clone display
- cloneheight — allows you to specify a height for the clone display
- clonerefresh — allows you to specify a refresh rate for the clone display

3.8.1 Extended Clone Mode CED Configuration

The following CED screenshot shows a sample Extended Clone mode setting configuration.



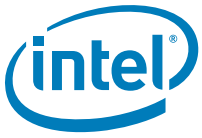
3.9 Gang DVO for the NS387R Transmitter

This release supports Gang DVO configuration for the National Semiconductor NS387R* transmitter. This feature allows a single display to receive data from two DVO ports (DVO-B and DVO-C). All attributes, DVO parameters, and user-defined DTDs are taken from the DVO-B port.

Gang mode is supported on the following chipsets:

- Intel® 852GM chipset
- Intel® 852GME chipset
- Intel® 855GME chipset

Note: In CED, you can enable Gang mode once the NS387R transmitter has been selected.



3.10 Scaling and Centering Configurations

This release supports the following scaling and centering configurations:

- Upscaling for the Chrontel CH7017/CH7308 LVDS Transmitters
- Internal LVDS Scaling With EDID Panels
- Alignment in Clone mode
- DVO as Primary
- Render Scaling modes to native panels connected to non-scaling port encoders

3.10.1 Upscaling for the Chrontel CH7017/CH7308 LVDS Transmitters

The IEGD can upscale lower-resolution modes (those smaller than the size of the respective panel) to the native size of the panel connected to a Chrontel CH7017* or CH7308* LVDS transmitter.

The IEGD uses a user-supplied DTD with the native flag set (also known as native DTD) as native timing for the panel connected to either a CH7017 or CH7308 transmitter.

If a native DTD is not supplied by the user, the IEGD takes the first available matching FP `info width` and `height` timings as native timing for the panel if standard timings were selected as part of `edid_avail` or `edid_not_avail` flags.

In order to support upscaling, the LVDS transmitters require the pipe to be set to native timing of the panel regardless of the user selected resolution. It also requires finding the native timing (also known as native DTD) of the panel based on user-supplied configuration information.

Both the CH7017 and CH7308 (SDVO) port drivers make the list of supported modes limited up to the size of panel. The port drivers also mark one of the timings as native DTD as follows (it goes to the next step only if native DTD isn't found in the current step).

1. It finds the timing with the user-defined DTD with the native DTD flag set. This becomes the native DTD for the panel.
2. If the panel is an EDID panel and user selected to use EDID DTDs, then the port driver marks the EDID DTD as native DTD.
3. If the user supplies a DTD without the native DTD flag set, then the port driver marks this one as the native DTD.
4. If none of the above steps works, the port driver finds the first matching timing for FP width, height and marks it as native DTD.

If none of the above steps work, then there is no native DTD and no upscaling is performed.

3.10.2 Internal LVDS Scaling With EDID Panels

The Internal LVDS connected to an EDID Panel supports scaling of modes other than native mode. To support this, the port driver exports information to the EDID parser that it can scale. The EDID parser does not remove other modes (that is, non-native modes) from the mode table. It only marks the native mode. When the IEGD queries the port driver on which modes are supported, the port driver then removes any modes that cannot be scaled (up or down depending on the port's hardware capability). When mode-setting occurs, the second display in Clone mode can indeed support non-native modes even though the panel had EDID. This occurs only if a native mode can be found the port driver can scale. Otherwise, the port driver ignores the scaling information and the IEGD proceeds normally.

The driver also supports Internal LVDS Scaling on EDID-less panels. The steps that enable this are the same as those described for the scaling of Chrontel LVDS transmitters in Section 3.10.1.

3.10.3 Centering Primary Display with Scaling Encoders

In Clone mode, the IEGD expects the primary display to have a framebuffer size (OS Aware mode) that matches the display's native size of panel timings. When a display is designated as the primary display in a Clone mode configuration, and the user wants the primary to be centered (as explained in Section 3.10.5), users may want this setup to align a primary display on a scaling encoder with a secondary one that can only center. This won't work by default for certain port encoders such as the internal LVDS, which default to hardware scaling. But IEGD has a mechanism to override hardware scaling thus forcing centering.

When possible, the IEGD allows centering of 640x480, 800x600, and 1024x768 resolutions on the primary display. In some cases (depending on panels), the image may appear on the top-left. It may also produce unusable output on some displays (such as a TV). Thus this type of configuration is more appropriate for LVDS panels.

To disable hardware scaling (and force centering for primary display on above modes), users only need to set the "Panel_Fit" attribute (attribute "0x12") to '0' (zero).

3.10.4 Enabling Render Scaling on Port Encoders Without Hardware Scaling

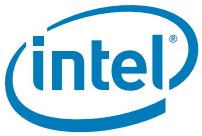
The IEGD Render Scaling feature allows the driver to support any one of the standard modes (640x480, 800x600, 1024x768 or 1280x1024) as a drawable framebuffer size output to a native panel and connected via a port encoder that doesn't hardware scale. To achieve this, the GPU engine repeats all rendering operations twice (from the original OS-targeted back buffer) to a separate front buffer, which is rendered via the 3D engine for scaling. This feature is enabled by turning on the "Panel-Fit" attribute (attribute 0x12) on a port driver that doesn't support that attribute. But this only happens if there is a native mode timing (see Section 3.10.1 for information about how native mode timing is determined).

Users should be aware that this feature can impact performance and produce scaled output which is inferior in quality to hardware encoder scaling.

3.10.5 Alignment in Clone Mode

In Clone mode, both can be configured with separate timings and different resolutions. The content is the same on both displays. In the case where resolutions are different on the cloned displays, the display identified as primary drives the display mode and framebuffer size. There are three options for the cloned displays in this situation:

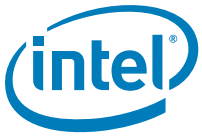
- *Panning*: If the clone display is smaller than the primary display, the displayed image can be off the screen with only the display showing a window into the overall image. Panning allows movement of the window, which is viewing the image based on the movements of the cursor.
- *Centering*: If the clone display is larger than the primary display mode, the display image can be centered in the clone display. Black borders are displayed around the image on the display.
- *Hardware Encoder Scaling*: This feature adjusts the resolution of the image from the primary display to fit the resolution of the clone display. This permits scaling up to a larger display (upscaling), or scaling down to a smaller display (downscaling). It also allows the full image to be displayed within the full resolution of the clone display.



Some systems may have cloned displays that cannot scale (for example, the NS387R transmitter) but have a primary display that can scale (such as an internal LVDS). In non-panning modes (i.e., centering/hardware scaling), this display combination will result in the primary display being scaled up (internal LVDS) but the clone display will be centered (NS387R). Section 3.10.3 explains how to force the primary display to center — thus allowing both displays to center. Or, *Render Scaling* as follows explains how to make both displays scale up to full size.

- *Render Scaling:* For clone display, a situation is possible where the primary display uses a hardware scaling port encoder and the secondary display uses a non-scaling port encoder. Assuming both displays are output via native panels, the resulting output should see the primary scaling of any smaller mode to full panel size. But the secondary display will center the smaller modes. The above explanation (see *Hardware Encoder Scaling*) explains how to align both displays to be centered. Using the Render Scaling feature, the opposite can be achieved. Ensure the non-scaling encoder is primary and enable Render Scaling on that port (see Section 3.10.4). This will make the GPU render-scale the smaller mode and achieve the full panel size. The clone display (now the scaling encoder) will, however, take the render-scaled image as its input (and output) to the clone display panel. This feature will be upgraded in the future so that the clone display can independently take in the original framebuffer image as its input.

This page is intentionally left blank.



4.0 VBIOS

4.1 Overview

The Intel Embedded Video BIOS incorporates many of the features and capabilities of the Intel® Embedded Graphics Drivers. The legacy VBIOS is still supported and is discussed in Chapter 8.0, "Legacy VBIOS." The 8.0 version of the VBIOS includes the following new feature:

- Support for Intel® Q35 Express chipset

4.2 System Requirements

The new Video BIOS can be built on a host Microsoft Windows* system and moved to the target system. The host system must have a 32-bit Microsoft Windows operating system installed with the capability to execute DOS commands from a command line window.

The target system must contain one of the following Intel chipsets:

- Intel® Q35 Express chipset
- Mobile Intel® GME965 Express chipset
- Intel® Q965 Express chipset
- Mobile Intel® 945GME Express chipset
- Intel® 945G Express chipset
- Intel® 915GV Express chipset
- Mobile Intel® 915GME Express chipset
- Mobile Intel® 910GMLE Express chipset
- Intel® 855GME chipset
- Intel® 852GME chipset
- Intel® 852GM chipset

The target system must contain a minimum of 64 Mbytes of RAM.

4.3 Configuring and Building the VBIOS with CED

The Intel® Embedded VBIOS is built with the Intel Configuration Editor (CED). The VBIOS will use the configuration that you specify in CED. The VBIOS is selected to be built when you specify the Video BIOS as a Target OS in your package configuration. After specifying the Video BIOS, follow all CED prompts, and be sure to select "Generate VBIOS" when available. The VBIOS will then be built when you select "Generate Installation" in CED.

Before building your VBIOS, you must set up your DOS environment with two items.

1. Install the Open Watcom* C/C++ compiler.
   The User Build System for the VBIOS relies on the Open Watcom C/C++ compiler to be able to build a 16-bit DOS binary required for the BIOS. The VBIOS has been tested with version 1.3 of the Open Watcom compiler. This can be downloaded from the following location:

   http://www.openwatcom.com
2. Set up directory paths.
   You must set up the PATH environment variable in DOS to be able to execute the Watcom compiler. If Watcom was installed with its default path, CED will by default be able to use it.

When you generate a VBIOS, the CED produces the following folders and files:

- Compiled_VBIOS folder
  - `iegdtsr.exe` (Terminate and Stay Resident executable)
  - `VGA.BIN` (Option ROM)
- `IEGD_X_x_VBIOS.zip` (this file is generated by the build system, where X is the major version and x is the minor version)

The `iegdtsr.exe` can be copied to any folder on the target machine. To run the TSR, boot the target machine with DOS, and then run the iegdtsr.exe from the DOS command line.

The `VGA.bin` file is the binary option ROM that can be merged with your system BIOS per the instructions provided by your system BIOS vendor.

The `IEGD_X_x_VBIOS.zip` file, where X is the major version and x is the minor version, contains default builds of the TSR executable and Option ROM for the various chipsets. The filenames are `iegdtsr-def.exe` and `vga-def.bin` and are located in the tsr or orom folder of the specific chipset folder (see Figure 9).

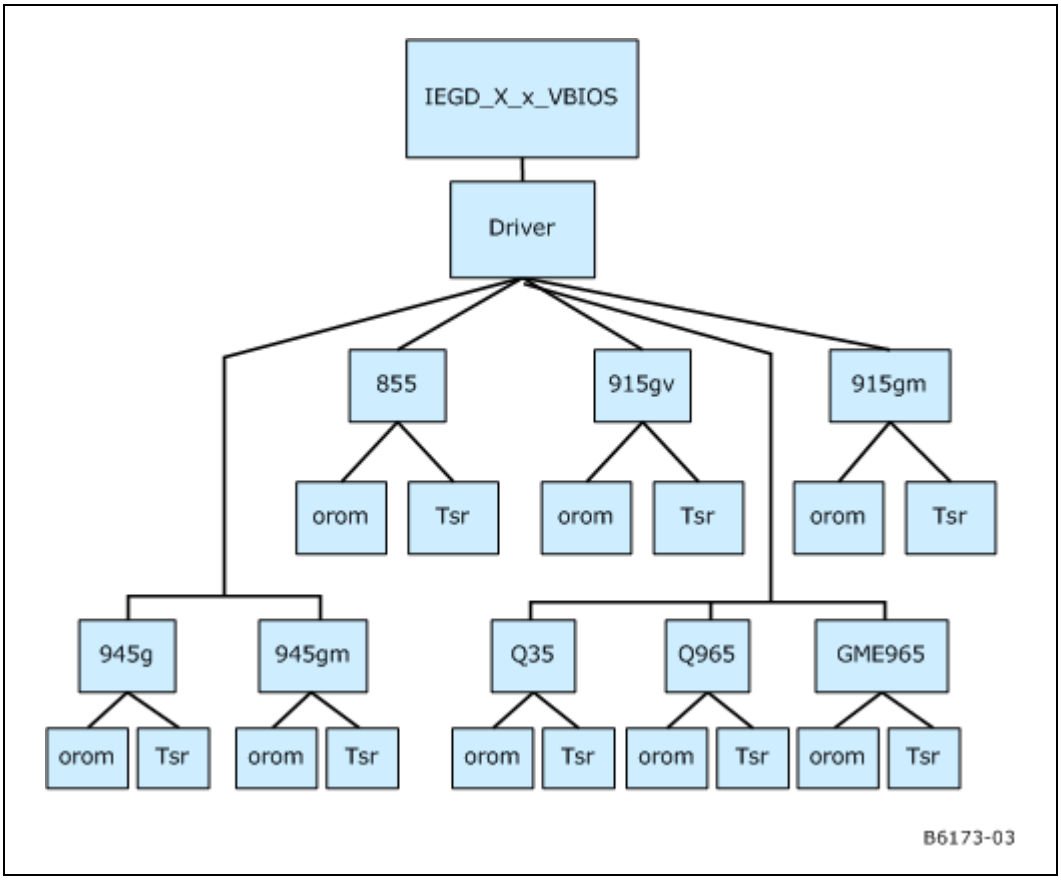
For further VBIOS build guidelines, see Section 4.3.3, "Building the VBIOS" on page 55.

4.3.1 Selecting the Build Folder

The 8.0 version of the VBIOS contains specific folders used for creating a VBIOS that is either an option ROM (OROM) that can be merged with the system BIOS, or an executable Terminate and Stay Resident (TSR) program for debugging purposes. There are also separate directories for the different chipsets that are supported. CED will build both the TSR and OROM.

Figure 9 shows the directory structure for the Video BIOS libraries contained within CED.

Figure 9. Video BIOS Directory Structure



4.3.2 Configuring the Video BIOS

CED is used to configure the VBIOS. The display settings will be used in the same manner as they are used for the driver.

4.3.2.1 COMMON_TO_PORT

This setting allows you to associate standard display names used in most system BIOSs to specific ports that are recognized by IEGD (e.g., LVDS, DVO-B, DVO-C). The VBIOS makes this association when the VBIOS calls the System BIOS Intel® 5F interrupt functions.

This setting is a six digit number, where each digit is associated with one of the system BIOS displays (from left to right):

1 : CRT - Standard analog CRT
2 : TV1 - TV Output 1
3 : EFP1 - DVI Flat Panel 1
4 : LFP - Local Flat Panel (Internal LVDS display)
5 : TV2 - TV Output 2
6 : EFP2 - DVI Flat Panel 2

The values above are an example of the typical displays and corresponding order used by a system BIOS. However, this may vary depending on how your system BIOS has implemented the displays and the Intel 5F interrupt functions.

The value in each position in the setting should be the associated port number. Using the typical settings above, if you want to associate CRT in the system BIOS with the internal CRT (port 5) and LFP in the system BIOS with internal LVDS (port 4) in the VBIOS, set COMMON_TO_PORT to be 500400.

Warning: This feature must be compatible with the system BIOS. If the system BIOS does not properly implement the Intel 5F functions, then using the `COMMON_TO_PORT` feature could cause unpredictable results with the displays. If you are unsure, set `COMMON_TO_PORT` to all zeros (000000) to disable this feature.

Note: Note that the `displaydetect` parameter must be set to `Enabled` in order for the `COMMON_TO_PORT` values to be used.

4.3.2.2 post_display_msg

This setting is a binary setting that enables (1) or disables (0) post messages to the display.

4.3.2.3 OEM Vendor Strings

The following settings are string values that allow you to set the values that are returned from the Intel 4F interrupt functions.

```
oem_string
oem_vendor_name
oem_product_name
oem_product_rev
```

4.3.2.4 Default Mode Settings

These settings establish the default VGA or VESA mode to use for the primary (0) and secondary (1) displays. The values should be set to a valid standard VGA or VESA mode (in hexadecimal format, for example, 0x117). Note that a VGA mode can only be set on one display and a second display is disabled unless the `DisplayConfig` parameter is set to `twin` or `clone` mode.

```
default_mode_0
default_mode_1
```

4.3.2.5 Default Refresh Settings

These settings allow you to specify which refresh rate is used for certain VESA modes on the primary and secondary displays. For example, mode 0x117 specifies refresh rates of 60 Hz, 75 Hz, and 85 Hz. This setting allows use to specify which of those three rates to use (specified in decimal, e.g., default_refresh_0=60).

```
default_refresh_0
default_refresh_1
```

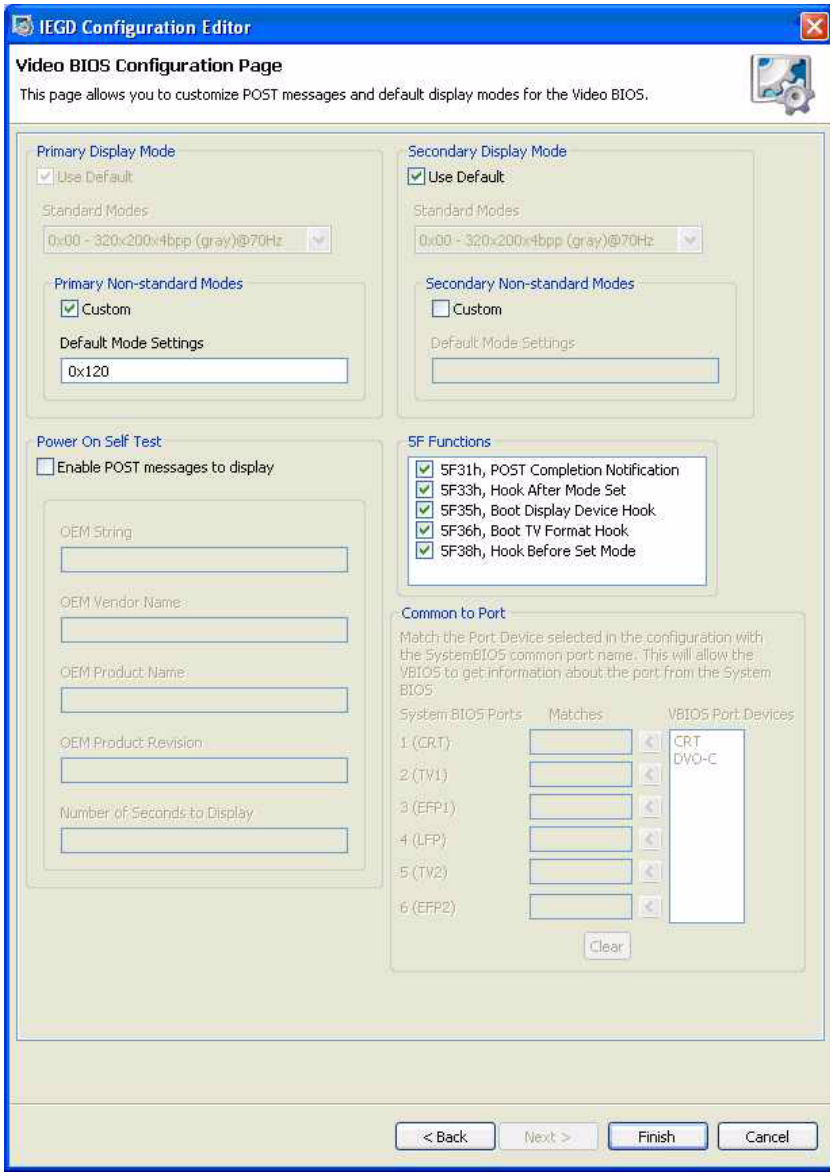
4.3.2.6 default_vga_height

This setting allows you to specify which resolution is used for certain VGA modes. Since only one VGA mode can be supported on both displays, this setting applies to the primary display mode (default_mode_0). For example, mode 3 specifies three possible resolutions: 640x200, 640x350, and 720x400. In this example, setting default_vga_height=350 indicates the resolution 640x350.

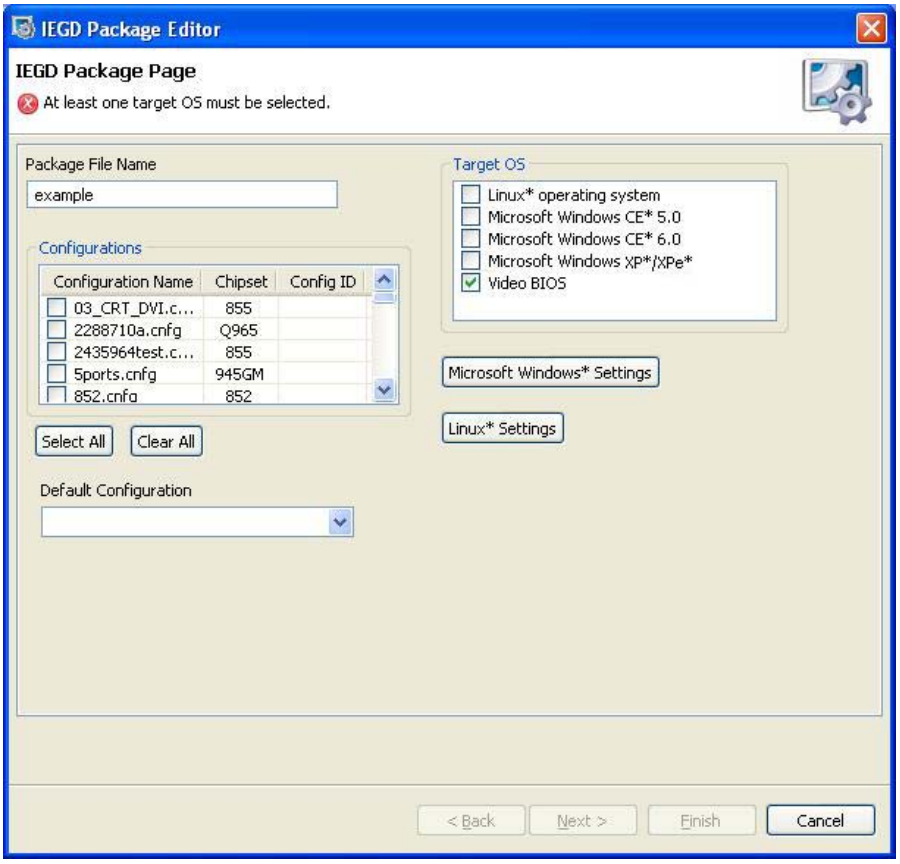
4.3.3 Building the VBIOS

CED is used to build the VBIOS. The following steps and screenshots outline a typical CED VBIOS build procedure.

1. Define your configuration via CED, being sure to complete the Video BIOS Configuration Page.

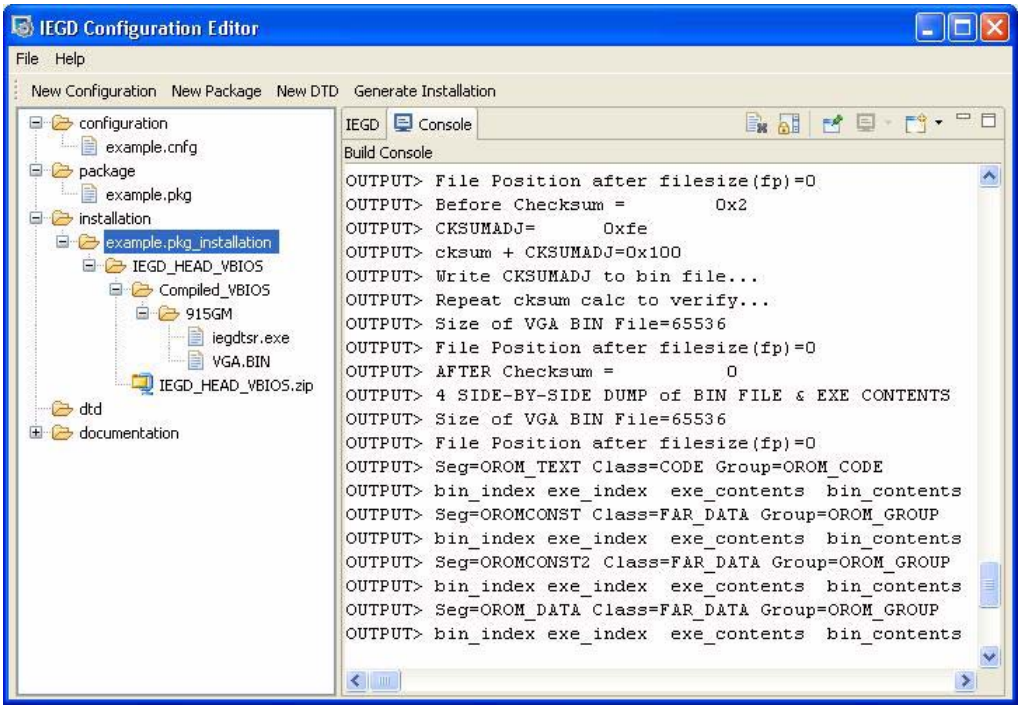


2. When you define the package you're building, be sure to select "Video BIOS" as "Target OS".



3. Generate the installation. The following may display:

4. Generated files should now be in your CED Installation folder.



4.4 VBIOS and Driver Compatibility

4.4.1 Data Dependencies Between VBIOS and Intel Graphics Driver

The Intel Embedded Graphics Drivers do not depend on any data from the VBIOS, and will either use driver settings or select default values for the attached displays. This allows the driver to properly operate with incompatible BIOS or BIOS replacements.

The Intel Embedded Graphics Drivers will retrieve settings, such as panel ID and other display settings from the Embedded VBIOS. The Embedded VBIOS allows for configuration of display timings that can also be used for the Intel Embedded Graphics Drivers.

4.5 VESA and VGA Video Modes

The VBIOS supports many VESA and standard VGA modes. Table 15 lists the modes and vertical refresh rates that are supported by the VBIOS.

Note: Although IBM labeled certain EGA modes with a (*) suffix and the VGA modes with a (+) suffix (such as mode 3, 3* and 3+), the VGA modes are so common that this document does not use the (+) suffix to refer to them.

The actual availability of any particular mode depends on the capabilities of the display device, the amount of memory installed, and other system parameters.

Table 15. Supported VGA Video Display Modes (Sheet 1 of 2)

Video Mode	Pixel Resolution	Color Depth (bpp)	Mode Type	Display Adapter	Font Size	Character Resolution	Dot Clock (MHz)	Horiz. Freq. (KHz)	Vert Freq (Hz)	Video Memory (KBytes)
00h	320 x 200	16 (gray) (4 bpp)	Text	CGA	8 x 8	40 x 25	25	31.5	70	256
	320 x 350	16 (gray) (4 bpp)		EGA	8 x 14	40 x 25	25	31.5	70	256
	360 x 400	16 (4 bpp)		VGA	9 x 16	40 x 25	28	31.5	70	256
01h	320 x 200	16 (4 bpp)	Text	CGA	8 x 8	40 x 25	25	31.5	70	256
	320 x 350	16 (4 bpp)		EGA	8 x 14	40 x 25	25	31.5	70	256
	360 x 400	16 (4 bpp)		VGA	9 x 16	40 x 25	28	31.5	70	256
02h	640 x 200	16 (gray) (4 bpp)	Text	CGA	8 x 8	80 x 25	25	31.5	70	256
	640 x 350	16 (gray) (4 bpp)		EGA	8 x 14	80 x 25	25	31.5	70	256
	720 x 400	16 (4 bpp)		VGA	9 x 16	80 x 25	28	31.5	70	256
03h	640 x 200	16 (4 bpp)	Text	CGA	8 x 8	80 x 25	25	31.5	70	256
	640 x 350	16 (4 bpp)		EGA	8 x 14	80 x 25	25	31.5	70	256
	720 x 400	16 (4 bpp)		VGA	9 x 16	80 x 25	28	31.5	70	256
04h	320 x 200	4	Graph	All	8 x 8	40 x 25	25	31.5	70	256

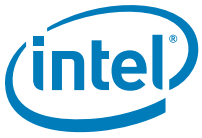


Table 15. Supported VGA Video Display Modes (Continued) (Sheet 2 of 2)

Video Mode	Pixel Resolution	Color Depth (bpp)	Mode Type	Display Adapter	Font Size	Character Resolution	Dot Clock (MHz)	Horiz. Freq. (KHz)	Vert Freq (Hz)	Video Memory (KBytes)
05h	320 x 200	4 (gray)	Graph	CGA	8 x 8	40 x 25	25	31.5	70	256
	320 x 200	4 (gray)		EGA	8 x 8	40 x 25	25	31.5	70	256
	320 x 200	4		VGA	8 x 8	40 x 25	25	31.5	70	256
06h	640 x 200	2	Graph	All	8 x 8	80 x 25	25	31.5	70	256
07h	720 x 350	Mono	Text	MDA	9 x 14	80 x 25	28	31.5	70	256
	720 x 350	Mono		EGA	9 x 14	80 x 25	28	31.5	70	256
	720 x 400	Mono		VGA	9 x 16	80 x 25	28	31.5	70	256
08h-0Ch	Reserved			-		-				
0Dh	320 x 200	16 (4 bpp)	Graph	E/VGA	8 x 8	40 x 25	25	31.5	70	256
0Eh	640 x 200	16 (4 bpp)	Graph	E/VGA	8 x 8	80 x 25	25	31.5	70	256
0Fh	640 x 350	Mono	Graph	E/VGA	8 x 14	80 x 25	25	31.5	70	256
10h	640 x 350	16 (4 bpp)	Graph	E/VGA	8 x 14	80 x 25	25	31.5	70	256
11h	640 x 480	2 (4 bpp)	Graph	VGA	8 x 16	80 x 30	25	31.5	60	256
12h	640 x 480	16 (4 bpp)	Graph	VGA	8 x 16	80 x 30	25	31.5	60	256
13h	320 x 200	256 (8 bpp)	Graph	VGA	8 x 8	40 x 25	25	31.5	70	256

Table 16 lists the VESA modes supported by the Video BIOS.

Table 16. VESA Modes Supported by Video BIOS (Sheet 1 of 3)

Video Mode	Pixel Resolution	Colors (bpp)	Mode Type	Display Adapter	Vertical Frequency (Hz)	Video Memory (Mbytes)
101h	640 x 480	256 (8 bpp)	Graph	VGA	60	0.5
	640 x 480	256 (8 bpp)	Graph	VGA	75	0.5
	640 x 480	256 (8 bpp)	Graph	VGA	85	0.5
103h	800 x 600	256 (8 bpp)	Graph	SVGA	60	1
	800 x 600	256 (8 bpp)	Graph	SVGA	75	1
	800 x 600	256 (8 bpp)	Graph	SVGA	85	1
105h	1024 x 768	256 (8 bpp)	Graph	XVGA	60	1
	1024 x 768	256 (8 bpp)	Graph	XVGA	75	1
	1024 x 768	256 (8 bpp)	Graph	XVGA	85	1

Table 16. VESA Modes Supported by Video BIOS (Continued) (Sheet 2 of 3)

Video Mode	Pixel Resolution	Colors (bpp)	Mode Type	Display Adapter	Vertical Frequency (Hz)	Video Memory (Mbytes)
107h	1280 x 1024	256 (8 bpp)	Graph	SXGA	60	2
	1280 x 1024	256 (8 bpp)	Graph	SXGA	75	2
	1280 x 1024	256 (8 bpp)	Graph	SXGA	85	2
111h	640 x 480	64K (16 bpp)	Graph	VGA	60	1
	640 x 480	64K (16 bpp)	Graph	VGA	75	1
	640 x 480	64K (16 bpp)	Graph	VGA	85	1
114h	800 x 600	64K (16 bpp)	Graph	SVGA	60	2
	800 x 600	64K (16 bpp)	Graph	SVGA	75	2
	800 x 600	64K (16 bpp)	Graph	SVGA	85	2
117h	1024 x 768	64K (16 bpp)	Graph	XVGA	60	2
	1024 x 768	64K (16 bpp)	Graph	XVGA	75	2
	1024 x 768	64K (16 bpp)	Graph	XVGA	85	2
11Ah	1280 x 1024	64K (16 bpp)	Graph	SXGA	60	4
	1280 x 1024	64K (16 bpp)	Graph	SXGA	75	4
	1280 x 1024	64K (16 bpp)	Graph	SXGA	85	4
112	640 x 480	16M (32 bpp)	Graph	VGA	60	2
	640 x 480	16M (32 bpp)	Graph	VGA	75	2
	640 x 480	16M (32 bpp)	Graph	VGA	85	2
115	800 x 600	16M (32 bpp)	Graph	SVGA	60	4
	800 x 600	16M (32 bpp)	Graph	SVGA	75	4
	800 x 600	16M (32 bpp)	Graph	SVGA	85	4
118	1024 x 768	16M (32 bpp)	Graph	XVGA	60	4
	1024 x 768	16M (32 bpp)	Graph	XVGA	75	4
	1024 x 768	16M (32 bpp)	Graph	XVGA	85	4

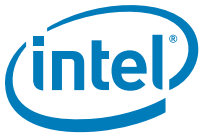
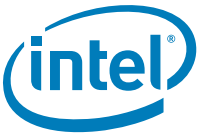


Table 16. VESA Modes Supported by Video BIOS (Continued) (Sheet 3 of 3)

Video Mode	Pixel Resolution	Colors (bpp)	Mode Type	Display Adapter	Vertical Frequency (Hz)	Video Memory (Mbytes)
11B	1280 x 1024	16M (32 bpp)	Graph	SXGA	60	8
	1280 x 1024	16M (32 bpp)	Graph	SXGA	75	8
	1280 x 1024	16M (32 bpp)	Graph	SXGA	85	8

This page is intentionally left blank.



5.0 Configuring and Installing Microsoft Windows Drivers

5.1 Overview

This section describes the driver-level information for the Microsoft Windows* operating system, which includes the following[1]:

- Microsoft Windows XP*
- Microsoft Windows XP Embedded*
- Microsoft Windows Embedded for Point Of Service* (WEPOS)

Note: Configuration and Installation information for the Microsoft Windows CE operating system is described in Chapter 6.0, "Installing and Configuring Microsoft Windows CE Drivers".

5.2 Configuration Information

5.2.1 Universal INF Configuration

Multiple display configurations can be specified in a single INF file. Each configuration is uniquely identified by the ConfigId parameter.

The driver reads the PanelId from the System BIOS during initialization and uses the configuration whose ConfigId matches the PanelId. If the System BIOS does not set a valid PanelId (for example, panelId = 0), the driver reads a configuration using ConfigId = 1. (A ConfigId value of 0 is invalid.)

You can override this behavior by specifying a ConfigId parameter as follows:

```
HKR,, ConfigId, %REG_DWORD%, %DEFAULT_CONFIG_ID%
```

In this case, the driver ignores the PanelId returned by the System BIOS. Instead, the IEGD uses the configuration information using the specified ConfigId.

5.2.2 INF File Backward Compatibility

The current version of the IEGD uses the new INF file format. You cannot use the new INF file with pre-5.0 versions of the IEGD. However, you can still use pre-5.0 INF file formats with the current version of the IEGD.

1. These versions of the drivers are not WHQL (Windows Hardware Quality Labs) certified.

5.2.2.1 INF File Backward Compatibility with IEGD Version 4.0

Version 4.0 of the IEGD provides backward compatibility with pre-4.0 versions of the INF file. This support is implemented through the PcfVersion key in the INF file, shown below:

```
HKR,, PcfVersion,    %REG_DWORD%, 0x0400
```

The IEGD uses this key to determine which version of the .inf file it is interpreting. When this key is present in the .inf file and its value is 0x0400, the driver reads it as a 4.0 .inf file. If this key is omitted from the .inf file or if its value is less than 0x0400, the driver reads the .inf file as a pre-4.0 file.

Note the following rules:

- If you use a pre-4.0 version of the .inf file with version 4.0 of the IEGD, the driver translates pre-4.0 configuration parameters to 4.0 parameters.
- You cannot use 4.0 parameters in a pre-4.0 .inf file. If you try, the driver ignores them.
- You cannot use pre-4.0 parameters in a 4.0 .inf file. If you try, the driver ignores them.

For example, the usestdtimings parameter is a pre-4.0 parameter. If it is specified in a 4.0 INF file, the driver ignores it. Similarly, if you attempt to add the edid_avail and edid_non_avail parameters to a pre-4.0 .inf file (that is, an .inf file where the PcfVersion key is not present), they are ignored by the driver.

The PcfVersion key is generated automatically by the CED utility and is placed in the [iegd_SoftwareDeviceSettings] section of the .inf file. The default iegd.inf file already contains the PcfVersion key. Please see Appendix A, “Example INF File” to view a sample .inf file.

5.2.3 Dual Panel Configuration

Below are the settings required to set the INF file to enable extended display configurations. Typically, these settings are output from the CED utility. However, the INF file may also be edited directly. See Table 17 for a description of these settings.

```
HKR, Config\%DEFAULT_CONFIG_ID%\General, DisplayConfig, %REG_DWORD%, 8
HKR, Config\%DEFAULT_CONFIG_ID%\General, PortOrder, %REG_SZ%, "5200"
```

5.2.4 Intel® 855GME Chipset Dual Display Example

The table below presents the dual display example for the Intel® 855GME chipset.

Table 17. Example of Intel® 855GME Chipset Dual Display Parameter Setting (Sheet 1 of 2)

Dual Display Combination	Port Order
CRT + Internal LVDS	“54000”
CRT + DVOB	“52000”
CRT + DVOC	“53000”
Internal LVDS + CRT	“45000”
Internal LVDS + DVOB	“42000”
Internal LVDS + DVOC	“43000”
DVOB + CRT	“25000”

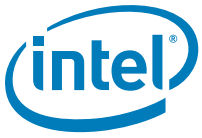


Table 17. Example of Intel® 855GME Chipset Dual Display Parameter Setting (Sheet 2 of 2)

Dual Display Combination	Port Order
DVOB + Internal LVDS	“24000”
DVOB + DVOC	“23000”
DVOC + CRT	“35000”
DVOC + Internal LVDS	“34000”
DVOC + DVOB	“32000”

5.2.5 Creating Registry Settings for Graphics Driver INF File

The driver settings are configured using CED. It generates the following output, which is then inserted into the graphics driver INF file before driver installation. CED simply translates the configuration options to the INF file. See Table 10, “Parameter Configuration Format” on page 26 for details on the specific settings and values, which also apply to the settings and values of the INF file. The values of the INF file may also be directly modified. See the example below for syntax and usage. Also, see Appendix A, “Example INF File” for a complete sample INF file.

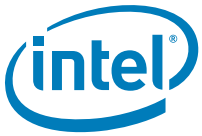
```
HKR,, PcfVersion,    %REG_DWORD%, 0x0700

HKR,, No_D3D, %REG_DWORD%, 0
HKR,, PortDrivers, %REG_SZ%, “ch7009 ch7017 fs454 lvds ns2501 ns387 sii164 ti410
th164”

;-------------------------------------------------------------------------------
[iegd_SoftwareDeviceSettings_nap]
HKR,, InstalledDisplayDrivers, %REG_MULTI_SZ%, iegddis
HKR,, MultiFunctionSupported, %REG_MULTI_SZ%, 1
HKR,, VgaCompatible, %REG_DWORD%, 0
HKR,, PcfVersion,    %REG_DWORD%, 0x0700

HKR,, No_D3D, %REG_DWORD%, 0

HKR,, PortDrivers, %REG_SZ%, “sdvo lvds tv”

;-------------------------------------------------------------------------------
[iegd_SoftwareDeviceSettings_gn4]
HKR,, InstalledDisplayDrivers, %REG_MULTI_SZ%, iegddis
HKR,, MultiFunctionSupported, %REG_MULTI_SZ%, 1
HKR,, VgaCompatible, %REG_DWORD%, 0
HKR,, PcfVersion,    %REG_DWORD%, 0x0700

HKR,, No_D3D, %REG_DWORD%, 0

HKR,, PortDrivers, %REG_SZ%, “sdvo lvds”

;===============================================================================
[Strings]

;-------------------------------------------------------------------
; Localizable Strings
;-------------------------------------------------------------------
Intel=”Intel Corporation”
DiskDesc=”Embedded Installation”

i830m=”830M Embedded Graphics Controller”
i835=”835 Embedded Graphics Controller”
i845=”845 Embedded Graphics Controller”
i855=”855 Embedded Graphics Controller”
i865=”865 Embedded Graphics Controller”
```

```
i915GD0="915G/915GV/910GL Embedded Graphics Controller Function 0"
i915GD1="915G/915GV/910GL Embedded Graphics Controller Function 1"
i915AL0="915GM/915GMS/910GML Embedded Graphics Controller Function 0"
i915AL1="915GM/915GMS/910GML Embedded Graphics Controller Function 1"
i945LP0="945G Embedded Graphics Controller Function 0"
i945LP1="945G Embedded Graphics Controller Function 1"
i945CT0="945GM Embedded Graphics Controller Function 0"
i945CT1="945GM Embedded Graphics Controller Function 1"

i965BW0="965G Embedded Graphics Controller Function 0"
i965BW1="965G Embedded Graphics Controller Function 1"
iG9650="G965 Embedded Graphics Controller Function 0"
iG9651="G965 Embedded Graphics Controller Function 1"
iQ9650="Q963/Q965 Embedded Graphics Controller Function 0"
iQ9651="Q963/Q965 Embedded Graphics Controller Function 1"
i946GZ0="946GZ Embedded Graphics Controller Function 0"
i946GZ1="946GZ Embedded Graphics Controller Function 1"
i965GM0="GM965 Embedded Graphics Controller Function 0"
i965GM1="GM965 Embedded Graphics Controller Function 1"
;------------------------------------------------------------------------
; Non Localizable Strings
;------------------------------------------------------------------------
SERVICE_BOOT_START     = 0x0
SERVICE_SYSTEM_START   = 0x1
SERVICE_AUTO_START     = 0x2
SERVICE_DEMAND_START   = 0x3
SERVICE_DISABLED       = 0x4

SERVICE_KERNEL_DRIVER  = 0x1

SERVICE_ERROR_IGNORE   = 0x0; Continue on driver load fail
SERVICE_ERROR_NORMAL   = 0x1; Display warn, but continue
SERVICE_ERROR_SEVERE   = 0x2; Attempt LastKnownGood
SERVICE_ERROR_CRITICAL = 0x3; Attempt LastKnownGood, BugCheck

REG_EXPAND_SZ = 0x00020000
REG_MULTI_SZ  = 0x00010000
REG_DWORD     = 0x00010001
REG_SZ        = 0x00000000
```

5.2.6 Dynamic Port Driver Configuration

The IEGD supports many third-party digital transmitters connected to the DVO or SDVO ports of the GMCH though device drivers called port drivers. These port drivers are dynamically loaded at startup. The driver configuration can be modified to add or remove availability of specific port drivers.

This section describes the portions of the `iegd.inf` file that can be modified to either add or remove a port driver for the Microsoft Windows version of the Intel® Embedded Graphics Drivers.



5.2.6.1 iegd.PortDrvs_xxx

The first step in either adding or removing a port driver is to identify the family of the chipset you are using. 852 and 855 are “Almador” graphics engine (alm), 915 and 945 are Napa (nap), and 965 is Gen 4 (gn4). Next locate the appropriate [iegd.PortDrvs_xxx] section for your graphics family. Below are the default settings for the blocks of associated port drivers for a particular graphics chipset family:

```
[iegd.PortDrvs_alm]
ch7009.sys
ch7017.sys
fs454.sys
lvds.sys
ns2501.sys
ns387.sys
sii164.sys
ti410.sys
th164.sys

[iegd.PortDrvs_nap]
sdvo.sys
lvds.sys
tv.sys

[iegd.PortDrvs_gn4]
sdvo.sys
lvds.sys
```

To remove one or more port drivers, delete the associated line from the `iegd.PortDrvs_xxx` block. To add a port driver, add the associated line into the appropriate `iegd.PortDrvs_xxx` block. For example, to add a new port driver for a device named “NewPD”, add the following line to the `iegd.PortDrvs_alm` block:

```
NewPD.sys
```

5.2.6.2 SourceDisksFiles

The next step to either add or remove a port driver is to identify the specific port driver file names in the `SourceDisksFiles` blocks. The default settings are as follows:

```
[SourceDisksFiles]
iegdmini.sys = 1
iegddis.dll  = 1
iegd3dg3.dll= 1
iegd3dg4.dll= 1
ch7009.sys  = 1
ch7017.sys  = 1
fs454.sys   = 1
lvds.sys    = 1
ns2501.sys  = 1
ns387.sys   = 1
sii164.sys  = 1
ti410.sys   = 1
th164.sys   = 1
sdvo.sys    = 1
tv.sys      = 1
```

To remove a port driver, delete the associated line in the [SourceDisksFiles] block. To add a port driver, add the associated line to the block. For example, to add a port driver for a device whose driver is named NewPD.sys, add the following line:

```
NewPD.sys  = 1
```

5.2.6.3 PortDrivers Registry Key

The next step is to modify the registry key in the appropriate [iegd_SoftwareDeviceSettings_xxx] section that defines the list of available port drivers. Below are the default values of this registry key in the iegd.inf file:

For the [iegd_SoftwareDeviceSettings_alm] block:

```
HKR,, PortDrivers, %REG_SZ%, "ch7009 ch7017 fs454 lvds ns2501 ns387
sii164 ti410 th164"
```

for the [iegd_SoftwareDeviceSettings_nap] block:

```
HKR,, PortDrivers, %REG_SZ%, “sdvo lvds tv”
```

For the [iegd_SoftwareDeviceSettings_gn4] block:

```
HKR,, PortDrivers, %REG_SZ%, “sdvo lvds”
```

Remove or add port driver names as appropriate to the list of port drivers specified within the quoted string. For example, to add support for a new port driver named “NewPD”, the registry key would be defined as follows:

```
HKR,, PortDrivers, %REG_SZ%, "ch7009 ch7017 fs454 lvds ns2501 ns387
sii164 ti410 th164 NewPD"
```

5.2.7 Creating an .sld file for Microsoft Windows XP Embedded Systems

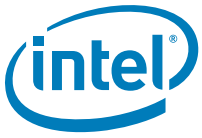
Microsoft Windows XP Embedded* operating systems require the use of an .sld (system level definitions) file. The following steps detail how to create such a file for IEGD from your custom iegd.inf file that you created using CED.

1. Run Component Designer.
2. In the **File** menu, select **Import.**
3. In the **Choose File for Import** dialog, select **Setup Information files (*.inf)**. in the **File of type** drop-down list.
4. Select **iegd.inf** from installation directory.
5. In the **Inf Processing Options** dialog, select **Automatic** in the **Parsing Options** dialog and click **OK**.
6. Click **Start** in the **Import File** dialog box. Close the dialog on completion. There should not be any errors.
7. If there are no errors, **Save** the .sld file.

5.2.8 Changing Default Display Mode

When installing the Intel® Embedded Graphics Drivers, Microsoft Windows selects a default display mode for the initial startup of the system. This is a 640 x 480 resolution in four-bit color mode.

In some cases, particularly with EDID-less LVDS displays, the 640 x 480 resolution may not be supported, so the default mode selected by Microsoft Windows must be changed. Otherwise, the display may not work after installation.



This default mode can be changed by adding the following registry keys to the `[iegd_SoftwareDeviceSettings]`section of the `iegd.inf` file:

```
HKR,, DefaultSettings.XResolution, %REG_DWORD%, 1024
HKR,, DefaultSettings.YResolution, %REG_DWORD%, 768
HKR,, DefaultSettings.BitsPerPel,  %REG_DWORD%, 32
HKR,, DefaultSettings.VRefresh,    %REG_DWORD%, 60
```

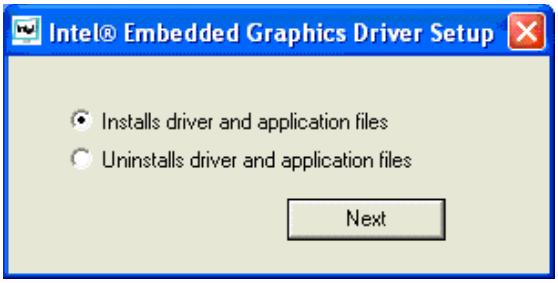
The example above makes the default resolution 1024 x 768, with a 32-bit color depth and a refresh rate of 60 MHz.

5.3 Installing the IEGD on Microsoft Windows

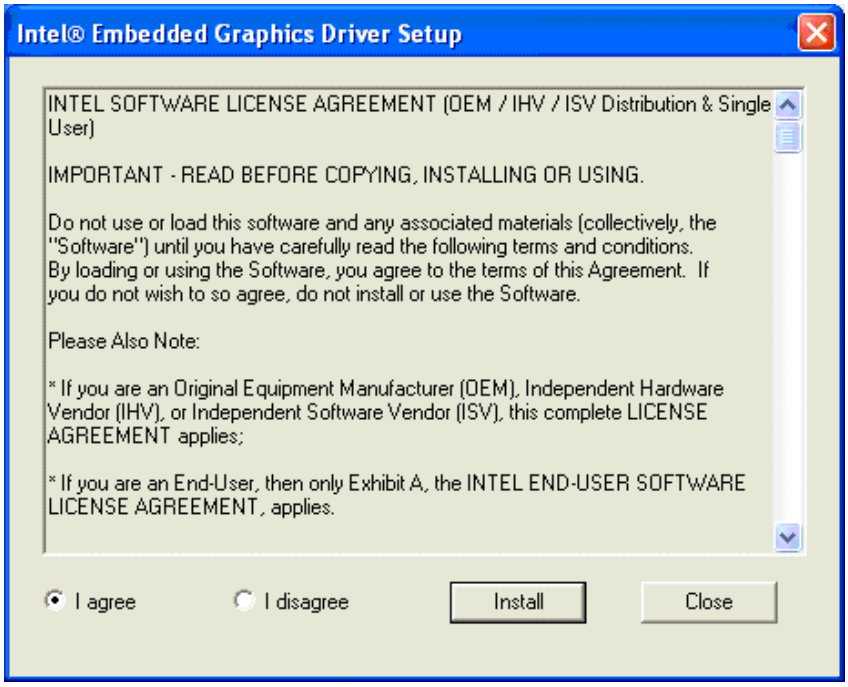
You can install and uninstall the IEGD on a Microsoft Windows system by using the `setup.exe` program located in the `Windows\Utilities` folder. The following procedure shows how to install the IEGD. Section 5.4, “Uninstalling the Current Version of the Driver” on page 71 provides instructions for uninstalling the current version of the IEGD.

Warning: If you have a previous version of the IEGD installed on your system, you must remove it using the Microsoft Windows **Add or Remove Programs** utility located in the **Control Panel**. Do not use the current version of the IEGD Install program to uninstall previous versions of the driver. If you do, unpredictable results may occur. You can use this program only to uninstall the driver from the current version. Each version of the driver has its own version of the installer/uninstaller utility.

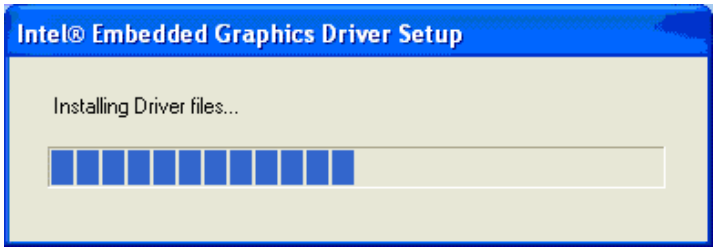
1. Double-click the `setup.exe` icon in the `Utilities` folder. The following dialog box appears.



2. To install the driver, make sure that **Installs driver and application files** is selected, and then click **Next**. The accept license screen appears.

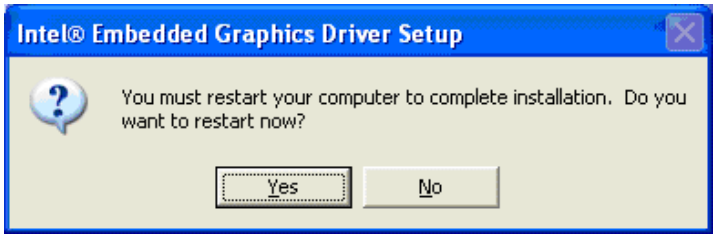


3. Click **I agree**, and then click **Install**. The installation begins and shows a progress bar as follows:



Note: If you get an "unsigned driver" warning, disregard and click **Continue** to allow the installation to continue.

4. After the driver and application files have been copied, the system must be restarted to complete the installation. If you want the installation program to restart your computer, click **Yes**.



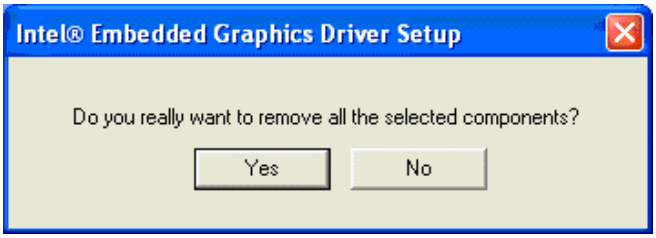
5.4 Uninstalling the Current Version of the Driver

You can use the `setup.exe` Microsoft Windows GUI program to remove the driver from your system. When you run the uninstaller program, it removes the following items from the system:

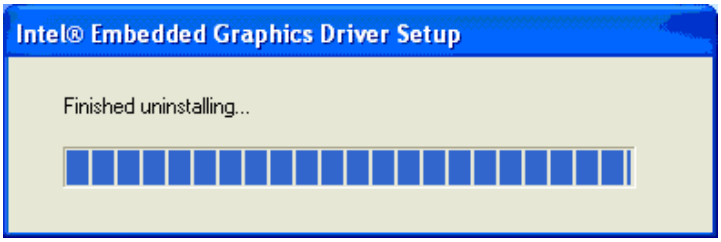
- The IEGD drivers
- The `.inf` and `.pnf` files from the `windows\system32\inf` folder.
- The `DisplayPage.dll` and `qt-mt332.dll` from the `windows\system32` folder
- Data registry items by running `regsvr32.exe` with the uninstall option.

Warning: You cannot use the **Add or Remove Programs** utility in the **Control Panel** to remove previous versions of the driver (each version of the driver has its own installer/uninstaller utility). To uninstall previous versions, click **Start ->Control Panel ->System ->Hardware ->Device Manager**, expand **Display adapters** then right-click the specific display adapter to be uninstalled, mouse-over and click **Uninstall**. Repeat if there's more than one previous version of the driver loaded.

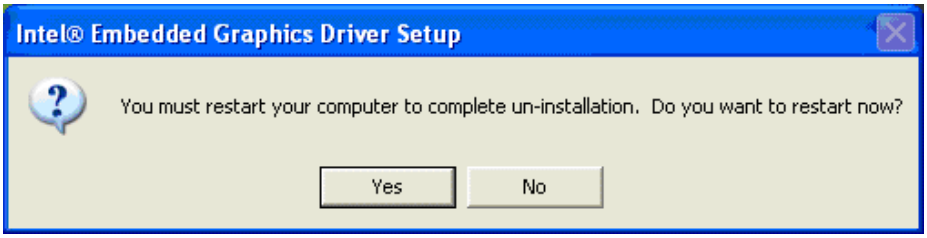
1. Click the `setup.exe` icon located in the `Utilities` subfolder of the `Windows` folder.
2. In the dialog box, select **Uninstalls driver and application files**, and then click **Next**. The following prompt appears:



3. Click **Yes** to remove the driver. A progress bar displays and when the driver has been removed, the following screen appears.



4. To complete the uninstallation, you must restart your system. If you want to restart your system now, click **Yes** in the following dialog box.



5.5 Run-Time Operation

Resolution, refresh rate, and color bit depth can be changed after installation and reboot via a Microsoft Windows display property sheet. On Microsoft Windows XP, extended desktop can be enabled and disabled, along with swapping primary and secondary displays. Other operations such as enabling and disabling ports (display output), rotation, port configuration, and attribute control are accessible via the standard display driver escape protocol.

5.6 Viewing and Changing the Driver Configuration from Microsoft Windows

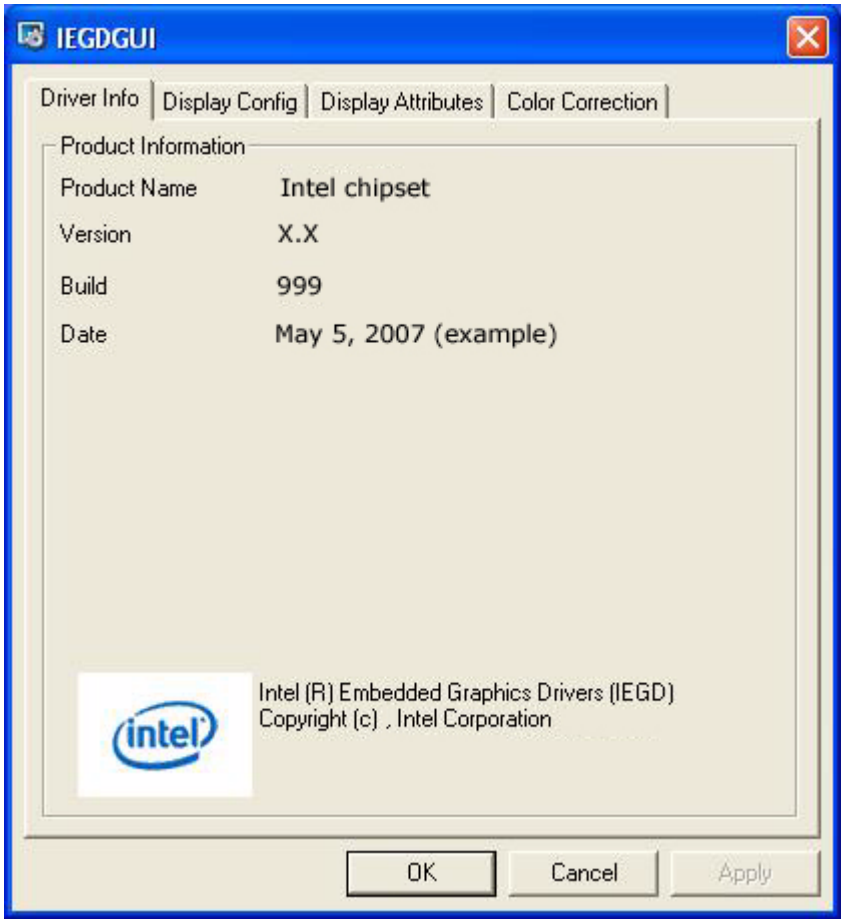
You can change certain configuration attributes of the IEGD using the `iegdgui.exe` program located in the `\Utilities` folder. On Microsoft Windows XP systems, you can access the IEGD Configuration tabs through the **Advanced Settings** tab of the **Display Properties** icon from the Windows dialog box. This program launches the IEGD Configuration GUI that consists of the following four tabs:

- **Driver Info** — Contains the driver information.
- **Display Config** — Contains current display information and allows configuration of display configurations, display resolutions and bit depth for primary and secondary displays, flip, rotation, and enabling/disabling for a given port.
- **Display Attributes** — Contains the supported Port Driver (PD) attributes and allows configuration of PD attributes.
- **Color Correction** — Contains color-correction information for the framebuffer and overlay. Using this tab, you can change the framebuffer and overlay color settings.

To view or change the driver settings using the GUI interface, follow this procedure.

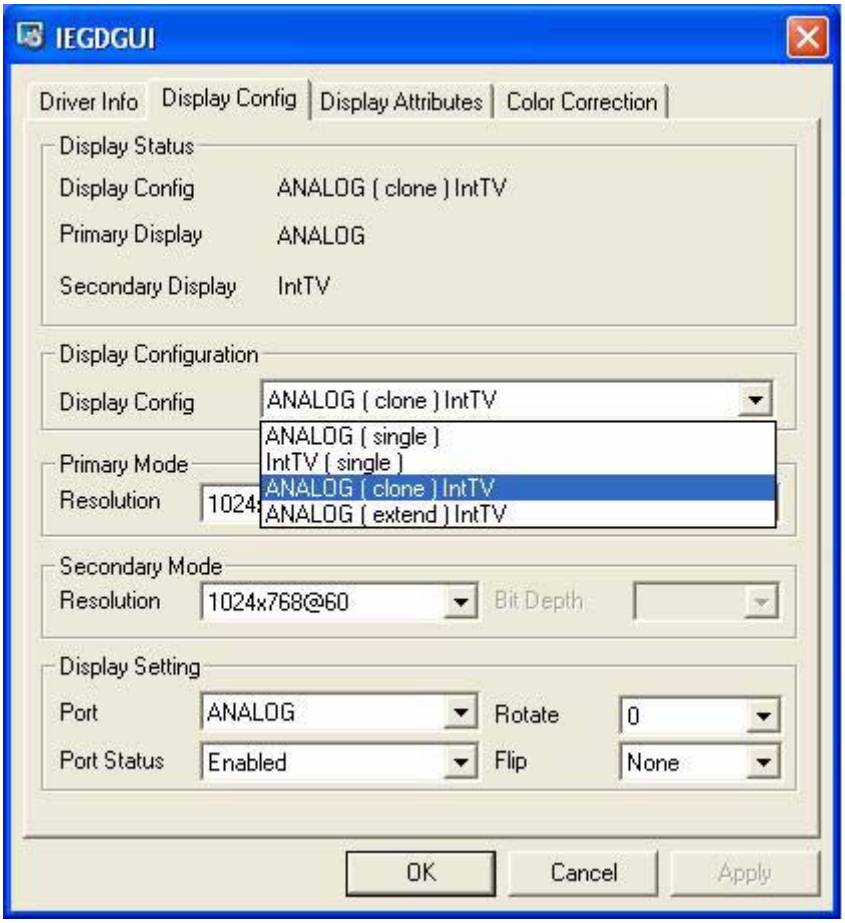
1. Double-click the `iegdgui.exe` icon in the `Utilities` folder. On Microsoft Windows XP systems, you can click the **Advanced Settings** tab in the **Display Properties** icon from the dialog box. The IEGD Configuration GUI Driver Info tab appears, showing information about the driver.

Figure 10. Example Runtime Configuration GUI — Driver Info Tab



2. Click the **Display Config** tab to show the current configuration.

Figure 11. Example Runtime Configuration GUI — Display Config Tab



The **Display Status** section of the above dialog shows the current configuration for the **Primary** and **Secondary** displays.

3. In the **Display Configuration** section of the dialog, select the required display configuration in the **Display Config** drop-down list. This allows the user to choose between Single, Twin, Clone and Extended for all connected ports. A maximum of two ports per display configuration is currently allowed.
4. In the **Primary Mode** and **Secondary Mode** sections of the dialog, change resolution and bit depth of the primary and secondary displays via the **Resolution** and **Bit Depth** drop-down lists.
5. In the **Display Settings** section of the dialog, view and change the settings for a port, rotate and flip the display via the appropriate drop-down lists:
   - **Port**: Allows you to select the required port.
   - **Port Status**: Allows you to enable or disable the selected port.
   - **Rotate**: You can rotate the display 0, 90, 180, and 270 degrees.
   - **Flip**: Inverts the display horizontally.

Note: If you change any configuration settings in the **Display Config** dialog box, click **Apply** for the changes to take effect.

6. Click the **Display Attributes** tab to view and change the attributes for a port. The screen that appears depends upon the port drivers used.

Figure 12. Example Runtime Configuration GUI — Display Attributes Tab

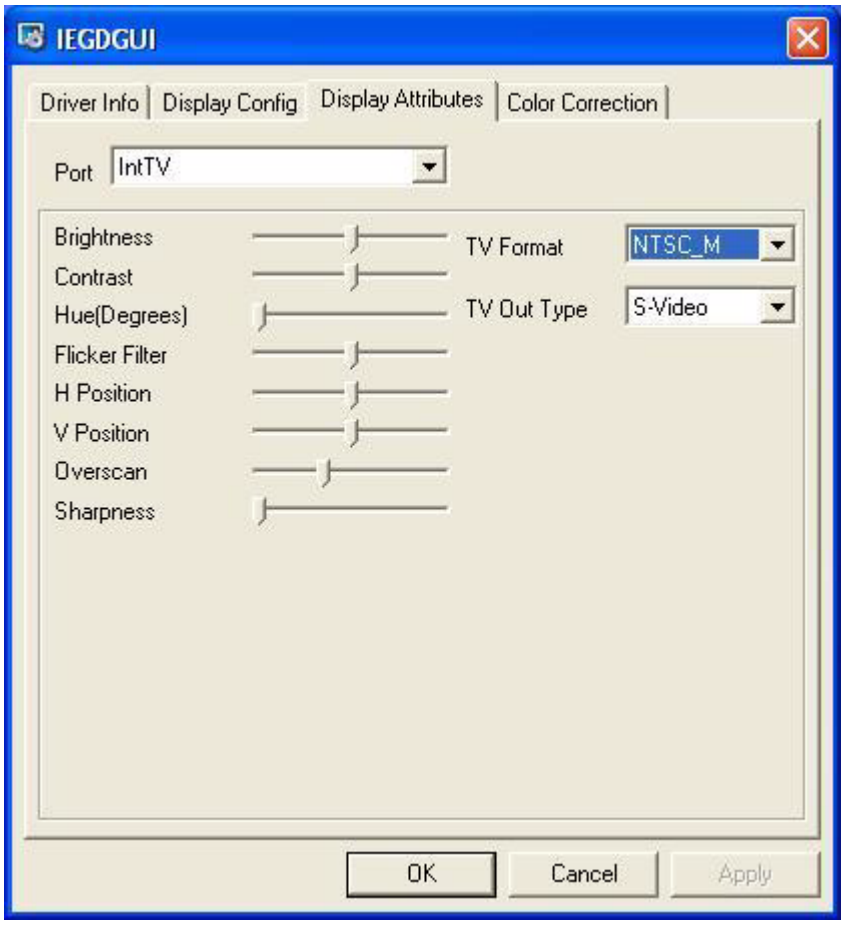


Figure 12 shows the attributes that can be changed for the selected port in the **Port** drop-down list. You can change the Port Driver by selecting the appropriate one for your device. The attributes that appear on this tab depend upon the selected port driver. Please see Appendix B, "Port Driver Attributes," for a complete list of port driver attributes.

7. Click the **Color Correction** tab to view and change color corrections. Figure 13 shows a sample Color Correction tab screen.

Figure 13. Example Runtime Configuration GUI — Color Correction Tab

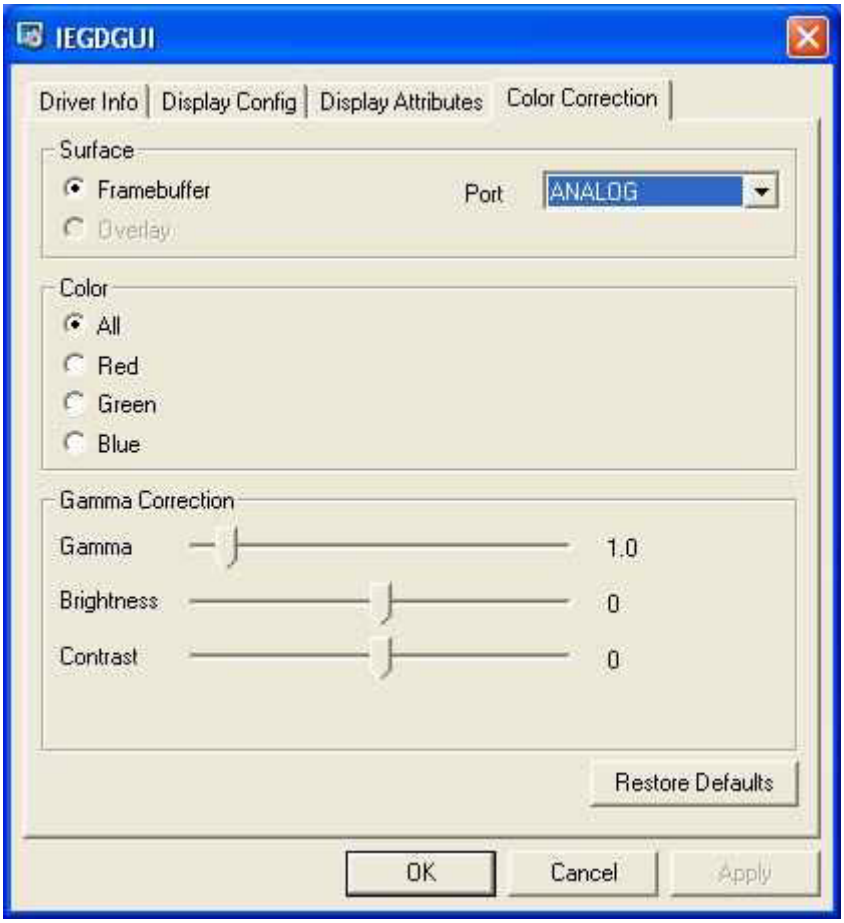
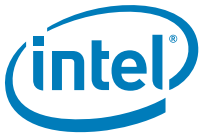


Table 18. Framebuffer Color Correction Values (applies to R, G, B color)

Gamma:	0.6 to 6.0 (default value is 1)
Brightness:	-127 to 127 (default value is 0)
Contrast:	-127 to 127 (default value is 0)

Table 19. Overlay Color Correction Values (applies to ALL color)

Gamma:	0.6 to 6.0 (default value is 1)
Brightness:	0 to 200 (default value is 100)
Contrast:	0 to 200 (default value is 100)
Saturation:	0 to 200 (default value is 100)



The following sub-steps present an example color-correction procedure:

a. Select **Framebuffer** in the **Surface** section and select the appropriate port for the color correction to be applied to or select **Overlay** in the Surface section for color correction to be applied to the overlay.

b. Select the required color to be corrected in the **Color** section.

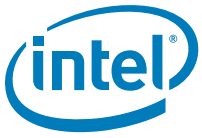
c. Select the required color attribute to be corrected in the **Gamma Correction** section.

d. Click **Restore Defaults** to restore the default values.

Note: If you make any changes to the color-correction settings, click **Apply** to have the changes take effect.

Note: The hardware does not support brightness, saturation, and contrast of the overlay and second overlay with RGB pixel format.

This page is intentionally left blank.



6.0 Installing and Configuring Microsoft Windows CE Drivers

6.1 Overview

This chapter describes the driver-level information for Microsoft Windows CE* operating systems.

6.2 Microsoft Windows CE Installation

The following sections describe how to install the IEGD on the Microsoft Windows CE 5.0 and 6.0 operating systems.

6.2.1 Prerequisites

The development system should have the following software installed:

- Microsoft Windows XP Professional, SP2
- Platform Builder for Microsoft Windows CE 5.0 or 6.0 (with latest service packs)

The target system must contain one of the following Intel chipsets:

- Intel® Q35 Express chipset
- Mobile Intel® GME965 Express chipset
- Intel® Q965 Express chipset
- Mobile Intel® 945GME Express chipset
- Intel® 945G Express chipset
- Intel® 915GV Express chipset
- Mobile Intel® 915GME Express chipset
- Mobile Intel® 910GMLE Express chipset
- Intel® 855GME chipset
- Intel® 852GME chipset
- Intel® 852GM chipset

The target system must contain a minimum of 64 Mbytes of RAM.

6.2.2 Integrating IEGD With Microsoft Windows CE Platform Builder

The integration/installation of the driver binaries depends upon the requirements of the target device; while ddi_iegd.dll is required, port drivers may be optionally included.

Note: Windows 6.0 does not employ the catalogue file mentioned below. For Windows CE 5.0, to integrate the driver binaries into the Platform Builder, the catalogue file included with the release should first be imported to the Platform Builder's catalogue (for details about how this is done, see Section 6.2.2.1). After the catalogue file is imported, find "Intel Embedded Graphics Driver" under Third Party -> Device Drivers -> Display and drag-and-drop it into the BSP.

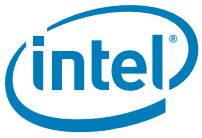
In order for the Platform Builder to use IEGD, the iegd.reg file included with the release has to be properly included into the BSP. For Windows CE 5.0, this means adding the following lines into the platform.reg file. Note that you must specify the correct path to the iegd.reg file.

```
;*******************************************************
;
;* IEGD Section for WinCE 5.0 Platform Builder
;*******************************************************
;
IF BSP_DISPLAY_IEGD

; @CESYSGEN IF CE_MODULES_PCI

; GWES will be able to auto-detect the display adapter if a candidate value
; points to the adapter's PCI instance. Values from Candidate1 to Candidate32
; (decimal) are possible; GWES will examine them sequentially until it finds
; a match.
[HKEY_LOCAL_MACHINE\System\GDI\DisplayCandidates]
       "Candidate6"="Drivers\\Display\\Intel"

[$(PCI_BUS_ROOT)\Template\IEGD]
"DisplayDll"="ddi_iegd.dll"
"Class"=dword:03
"SubClass"=dword:00
"ProgIF"=dword:00

"VendorID"=multi_sz:"8086", "8086", "8086", "8086", "8086", "8086", "8086", "8086",
"8086", "8086", "8086", "8086", "8086", "8086", "8086", "8086", "8086", "8086",
"8086", "8086", "8086", "8086", "8086", "8086", "8086"

"DeviceID"=multi_sz:"3582", "2572", "2562", "357B", "3577", "1132", "7125", "7123",
"7121", "2582", "2782", "2592", "2792", "2772", "2776", "27A2", "27A6", "2982",
"2983", "29A2", "29A3", "2992", "2993", "2972", "2973"

; include the path to the iegd.reg file in the release package
#include <specify_path_here>\iegd.reg

; @CESYSGEN ENDIF CE_MODULES_PCI
ENDIF ; BSP_DISPLAY_IEGD
```

Finally, to include the actual driver binaries into the OS image, you must reference them in the BSP's BIB file by appending the path to ddi_iegd.dll and the port drivers into platform.bib, as shown below.

Figure 14. Sample FILES Block from platform.bib File

```
FILES
;       Name            Path                                Memory Type
;       --------------  ----------------------------------  -----------

;       @CESYSGEN IF CE_MODULES_DEVICE
;       @CESYSGEN ENDIF CE_MODULES_DEVICE

        ddi_iegd.dll    <specify_path_here>\ddi_iegd.dll     NK
        ch7009.dll      <specify_path_here>\ch7009.dll       NK
        ch7017.dll      <specify_path_here>\ch7017.dll       NK
        fs454.dll       <specify_path_here>\fs454.dll        NK
```



6.2.2.1 Catalogue Feature File

For Windows CE 5.0, IEGD's Catalogue Feature File, `iegd.cec`, is provided in the release package. To import IEGD into the workspace's catalogue, complete the following steps:

1. From the File menu, select **Manage Catalog Features**.
2. Choose **Import**.
3. In the **Import Catalog Features** dialog box, select the `.cec` file, and then click **Open**.
4. From the **View** menu, select **Catalog** to display the Catalog.

6.3 Microsoft Windows CE Configuration

The following sections describe how to configure the IEGD on the Microsoft Windows CE operating system. All the IEGD-specific registry keys are located within the path `[HKEY_LOCAL_MACHINE\DRIVERS\Display\Intel]`

All keys use one of the following syntax:

`"<keyname>"=dword:<value>`,

or

`"<keyname>" = <value>`

where `<value>` in the second case is a string in double quotes.

Note: Unless specified otherwise, the "value" field is in hex format.

The `iegd.reg` file contains display configuration registry entries for the IEGD. A sample `iegd.reg` file is provided along with the driver package. The content of this file may be included through the "`#include`" directive in `platform.reg` (see Section 6.2.2), or it may be copied into the proper section in `platform.reg`.

6.3.1 Basic Driver Configuration

This section discusses basic driver configuration keys located in `[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\General]`.

Table 20 lists the keys in the "`Intel`" folder.

Table 20. [HKLM\DRIVERS\Display\Intel] Registry Keys

Registry Entry	Description	Possible Ranges
PCFversion	Specifies the version of the current configuration file.	400 or 700
ConfigId	This selects the configuration set.	1, 2, 3, 4, or 5
PortDrivers	List of port drivers to be dynamically loaded when the system boots. The dll's must exist in the `C:\Windows` directory. DVO transmitter port drivers to load when the system boots.	Space separated string enclosed in quotes, where each port driver name is listed in the string. The default string included with the release has all supported port drivers.

6.3.1.1 Graphics Memory Configuration

The Intel Embedded Graphics Suite (IEGS = VBIOS + Graphics driver) provides the ability to dedicate additional memory for graphics functions on the Microsoft Windows CE platform. This is known as *reserved memory*. The amount of reserved memory is selected by firmware. The reservation size is passed to the graphics driver through a scratch register available on the GMCH. Reserved memory is useful in minimizing the amount of memory stolen from the OS for memory-limited, embedded systems. For instance, if firmware utilizes a 640 x 480, 32-bit framebuffer, a total of 1.2 Mbytes is required. Stolen memory would need to be configured as 8 Mbytes or higher, since the next smaller option is only 1 Mbyte, too small for the 640 x 480, 32-bit framebuffer. In such a case, stolen memory can be programmed to 1 Mbyte. The additional memory required for the framebuffer can then be provided by reserved memory, allowing a minimum amount of memory to be removed from the OS.

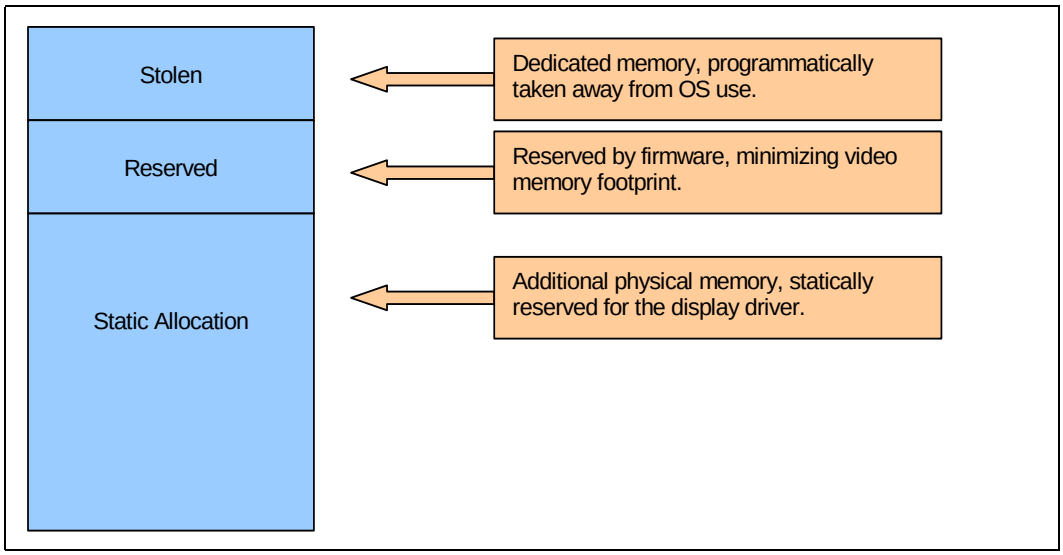
Note: Reserved memory is only available on the Microsoft Windows CE operating system, and must be accounted for in the `config.bib` memory layout file.

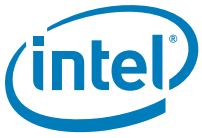
Additionally, the Microsoft Windows CE display driver can be configured for either static or dynamic allocation of video memory. The static model preallocates physical memory for the display driver and provides a more efficient surface allocation scheme. The dynamic model allocates surface memory on demand from the system and will incur a small performance hit. However, the dynamic model has the advantage of deallocation of video memory when not required, thus making it available to other applications.

The static memory model requires a base and size specification registered in the `project.reg` file. The base + size must reach to top of memory (TOM). Since this is not required to be specified in the config.bib memory map, care must be taken not to overlap any other memory arenas with the static allocation. See Section 6.2, “Microsoft Windows CE Installation” on page 79 for further details on how to configure the static memory model.

Figure 15 shows a typical memory map, using a static memory model.

Figure 15. Typical Memory Map Using Static Memory Model





6.3.1.2 Defining Graphics Memory Size

The IEGD supports static as well as dynamic location of graphics memory. For static video memory, the two registry settings ReservedMemoryBase and ReservedMemorySize denote where and how big this area will be. This portion of memory will be taken care off by the built in GART driver. The size should include the stolen memory (BIOS setting, if applicable).

If either of these registry settings are zero or not defined in the registry, then dynamic memory allocation is used for video memory management. The usage will be dependent on system resources. This is in line with the Dynamic Video Memory Technology.

As an example of static video memory, if the stolen memory is 8 Mbytes and you want a total of extra 56 Mbytes of graphics memory for a total of 64 Mbytes of graphics memory you would want to have these settings: (On a 128 Mbyte machine)

```
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\General]
       "ReservedMemoryBase"=dword:4000000
       "ReservedMemorySize"=dword:4000000
```

Which means that the managed graphics memory pool will begin at physical address 0x4000000 (64 Mbytes) and is 64 Mbytes of size.

Note: Always remember to include the amount of stolen memory in this number.

The lower 64 Mbytes in the above machine will be used for the NK.bin and RAM, so therefore you must change your `config.bib` accordingly. The configuration for the given machine is (debug build):

```
NK          80220000  01650000  RAMIMAGE
RAM         81700000  02790000  RAM
```

In this configuration, the NK.BIN image takes 22.3125 Mbytes and the rest of 41.6875 Mbytes is RAM. The release build should have much smaller NK area. For better control, users may turn AUTOSIZE=OFF to calculate exactly where the image and RAM starts and ends followed by the location of the video memory pool with no possibility of runtime deviation.

These two settings, together with the registry settings mentioned above, will determine the memory layout.

The example `config.bib` and the memory configuration settings in the `iegd.reg` are validated settings that can be used for static video memory of 16 Mbytes in size starting from the 48 Mbytes address, thus including the stolen memory.

6.3.1.3 Framebuffer and Video Surface Size

Two additional optional registry settings are available to limit the framebuffer size of the display driver and the total size of offscreen video surfaces.

The `MaxFbSize` registry entry will control the maximum size of the framebuffer only. Actual usage will depend on the mode being used.

The `PageReqLimit` registry entry will control the total size in pages (4 Kbytes) of all video surfaces, buffers allocated for any use. Both of these registry configurations apply to both the static as well as dynamic video memory management explained in the previous section. The default below indicates that a maximum of 2 Mbytes are used for the framebuffer and a maximum of 16 Mbytes are permitted for all offscreen videosurface allocations.

```
"MaxFbSize"=dword:200000
"PageReqLimit"=dword:1000
```

In the case of Microsoft Windows CE, because the OS does not allow for dynamically setting the framebuffer size, the MaxFbSize can be changed to match the mode setting being used in order to minimize on video memory wastage. The following are different suggested values for MaxFbSize for different display modes. These values have not been validated. Note that 640x480 is calculated as 640x512 and 800x600 is calculated as 800x768 for stride alignment purposes.

```
640x512X16 = A0000
640x512X24 = F0000
640x512X32 = 140000
800x768X16 = 12C000
800x768X24 = 1C2000
800x768X32 = 258000
1024x768X16 = 180000
1024x768X24 = 240000
1024x768X32 = 300000
```

6.3.1.4 Video Surface Allocation Rule

Another two optional registries entries determine a minimum width and height dimension that allows video surface allocations to succeed.

In the Microsoft Windows CE GDI, video surface allocations can happen with a REQUIRE_VIDEO_MEMORY or a PREFER_VIDEO_MEMORY flag. The following options force surface allocations with the PREFER_VIDEO_MEMORY flag to be allocated in system memory if the width and height are lower then what is stated in the following entries.

```
"MinVidSurfX"=dword:10
"MinVidSurfY"=dword:10
```

In this example, surfaces allocated with the PREFER_VIDEO_MEMORY where the width and height are both less than 16 pixels are forced to be in system memory.

6.3.1.5 System to Video Stretch Blit

System to Video Memory stretch blits are not natively supported on Intel GMCH devices. This feature allows you to enable a soft copy of system surfaces to video surfaces in order to conduct an accelerated stretch blit. The advantage of this is that the stretch blit then utilizes the blend engine and hardware filtering can be applied. The filtering options are listed in Section 6.3.1.7.

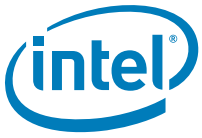
A value of 1 for the “SysToVidStretch” enables system to video stretch blits, as described above, while a value of 0, disables this feature and forwards all system to video stretch blits to the emulator provided by the operating system.

```
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\General]
"SysToVidStretch"=dword:0
```

6.3.1.6 iegd.reg File Backward Compatibility

The Intel Embedded Graphics Driver expects a configuration file in the PCFVersion 700 format. However, the driver will maintain backward support with version 4.0. This support is implemented through the PcfVersion key as shown below:

```
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\General]
"PcfVersion"=dword:400
```



The IEGD uses this key to determine the format of the configuration file. When this key is present, IEGD parses the configuration file using the format specified by the key (400 or 700). If this key is not present, then IEGD assumes 4.0 format.

6.3.1.7 Blend Filtering

The blend filtering method can be selected via the `BlendFilter` registry setting. The filter method chosen will be used for all stretch blit operations, including video to video stretch blits, as well as system to video stretch blits. The default filtering method is `Nearest`.

Available Blend Filter options:

```
0 == Default == Nearest
1 == Binlinear
2 == Anisotropic
3 == 4x4
```

```
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\General]
“BlendFilter”=dword:2
;  Blend filter to use for all stretch blits
;  BlendFilter 0 == NEAREST
;  BlendFilter 1 == BILINEAR
;  BlendFilter 2 == ANISOTROPIC
;  BlendFIlter 3 == 4X4
```

6.3.2 Configuration Sets

The Intel® Embedded Graphics Drivers allows multiple configuration sets for OEMs who wish to use the same iegd.reg file across different platforms. There can be up to 16 instances of configurations. The registry key described in the previous section, ConfigId, ensures the display driver selects the right instance. Each instance may contain multiple groups of per-config and per-config+per-port platform customizations.

The configuration sets are defined in the registry tree as

```
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\<platform>\<config id>],
```

Where 'config id' is the configuration number. The “ConfigID” key described in the previous section selects the active configuration set.

6.3.3 General Configuration

Registry keys described in this section can be found in `[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\<platform>\<config id>\]`, where 'config id' is the configuration number, and where `<platform>` is one of the following: ALL, Q35, GME965, Q965, 945G, 945GME, 915GV, 915GME, 910GMLE, 855GME, 855GM, 852GME, and 852GM. The driver first attempts to find the configuration or platform on which it is booted, but if the configuration for that platform is not present, the driver uses the ALL platform setting.

Table 21. [HKLM\Drivers\Display\Intel\<platform>\<config id>\]Registry Keys (Sheet 1 of 2)

Registry Entry	Description	Possible Ranges
Width	Width of the display	Width and Height must be expressed as hexidecimal values. For example: 1024 x 768: 400 x 300 800 x 600: 320h x 258 640 x 480: 280 x IE0
Height	Height of the display	See above.
Depth	Color depth in bpp (bits per pixel)	Depth must be expressed as a hexidecimal number and must be one of the following values: 8bpp: 8 16bpp: 10 24bpp: 18 32bpp: 20 (Note that the Intel 855 and 915 chipsets do not support 24 bpp.)
Refresh	The refresh rate of the display.	Refresh rate must be in hex: 60 : 3c 70 : 46 75 : 4b 85 : 55 etc... This value can be any valid refresh rate as long as the display port supports it. A refresh of '0' takes the first refresh that matches `width`, `height` and `depth`.
NO_D3D	Specify whether to enable D3D. ***Note:*** For Windows CE 5.0, this registry entry must be set to 1 as IEGD currently does not support D3D Mobile on Windows CE 5.0 systems.	0 = Enable D3D 1 = Disable D3D Default is 0.
ReservedMemoryBase ReservedMemorySize	Video memory can be statically reserved or dynamically allocated on demand. If both *ReservedMemoryBase* and *ReservedMemorySize* are non-zero, then Video memory allocation utilizes the static model.	The ReservedMemoryBase plus the ReservedMemorySize must extend to the TOM (Top Of Memory) and not conflict with other reserved memory arenas in config.bib. Default for both base and size is zero, indicating a dynamic allocation model. Default behavior disables static memory model.
MaxFbSize	Maximum size of the expected framebuffer. By providing this hint, the display driver can more efficiently organize GART memory, leading to a smaller video memory consumption.	Must be greater than or equal to the expected size of framebuffer. Units are in bytes. Specifying zero causes the default framebuffer reservation sizing. Default: All other chipsets: 16 Mbytes
MinVidSurfX MinVidSurfY	In pixels, the minimum width and height of surfaces in order to be acceptable for allocation in Video memory. Due to hardware restrictions that optimize memory access, it is advisable to reserve video memory for larger surfaces and allow GDI and DirectDraw* to allocate small surfaces from system memory.	No limitations. Suggested values for both width and height are 10. Default value for both width and height is 1. Default: MinVidSurfX = 1 MinVidSurfY = 1

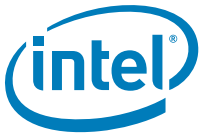


Table 21. [HKLM\Drivers\Display\Intel\<platform>\<config id>\]Registry Keys (Sheet 2 of 2)

Registry Entry	Description	Possible Ranges
SysToVidStretch	Enables system-to-video memory stretch blit operations to take advantage of hardware-accelerated filtering. Normally, it is more efficient to allow GDI to conduct system-to-video stretch blits, but the default filtering used by GDI is Nearest. See the *BlendFilter* key below for hardware accelerated filtering options.	0 = Disabled 1 = Enabled Default: 0
BlendFilter	Provides selection of hardware-accelerated filtering methods for stretch blit operations.	0 = Nearest 1 = Bilinear 2 = Anisotropic Default: 0
TearFB	If enabled, all blit operations to the framebuffer are synchronized with video sync to eliminate any visible tearing or flickering on the display screen. Disabling this feature achieves a performance gain.	0 = Disabled, tearing allowed 1 = Enabled, no visible tearing Default: 0
DisplayConfig	The “DisplayConfig” key sets the display configuration to be in Single, Twin, Clone, or Vertical Extended modes. (Unlike Microsoft Windows XP, Microsoft Windows CE does not support Extended mode). Does not, however, dictate what type of display ports will be used.	1 (single), 2 (clone), 4 (twin), 5 (vertical extended)
DisplayDetect	The “DisplayDetect” key allows the user to enable a display port only if a display device is connected. Displays without EDID will not be detected.	0 = disable 1 = enable Default: 0
PortOrder	The PortOrder setting ensures the correct display port types are used based on user selection.	See Section 6.3.3.1.

6.3.3.1 PortOrder Information

PortOrder specifies the actual ports that are used for the Primary and Secondary display. As shown in Table 22, the port numbers are slightly different among the supported chipsets.

Table 22. PortOrder Information

Port Number	85x Chipsets	91x Chipsets
1	DVO A Port	Integrated TV Encoder
2	DVO B Port/RGBA Port	DVO B Port/RGBA Port
3	DVO C Port	DVOC Port
4	Internal LVDS port	Internal LVDS Port
5	Analog Port	Analog Port

The driver attempts to use the ports in the order specified by “PortOrder”. For example, “PortOrder” = “5420” will assign the analog port to the primary display and the LVDS port to the secondary display (if any), assuming all the ports are present and detected. Suppose port “5” is not present, in that case the driver tries to assign the next port (4, in this case) in line to the primary display, resulting LVDS port for primary and DVO B port for secondary.

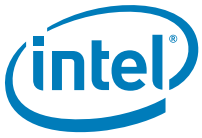
Setting PortOrder to “00000” causes the driver to use default internal settings.

```
*************

[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\General]
;-----------------------------------
; Select Port Order
;-----------------------------------
“PortOrder”=“54320”

;  PortOrder specifies the actual of port
;  that will be taken for the Primary /
;  Secondary ports if there are duplicates
;  of the same type. For example, if both
;  Primary and Secondary are digital, then
;  port order will which DVO ports will be
;  first and second. The section below gives
;  the port order numbers for various chipsets.
;  Specify value “0000” to use default settings.
;  On i915 chipsets:
;  ================================
;    1 - Integrated TV Encoder
;    2 - DVO B port/RGBA port
;    3 - DVO C port
;    4 - Internal LVDS port
;    5 - Analog port
;
;  On 85x chipsets:
;  ================================
;    1 - DVO A port
;    2 - DVO B port/RGBA port
;    3 - DVO C port
;    4 - Internal LVDS port
;    5 - Analog port
;
```

6.3.3.2 Vertical Extended Mode

The Windows CE IEGD driver supports Vertical Extended display mode, which is one large framebuffer that extends across two displays by doubling the height of resolution. The top half of the framebuffer is on the first pipe and the bottom half is on the second pipe. The Windows CE operating system is unaware of the two displays. This feature is supported only on the dual-pipelined chipsets, which is every supported platform stated in Section 6.2.1.

This feature is enabled through the `DisplayConfig` key in the `project.reg` file. The resolution, bit depth, and refresh rates of both displays must be the same. Vertical and horizontal panning are *not* supported. DirectDraw is supported on both pipes, but DirectDraw 3D must be disabled when Vertical Extended Display mode is enabled.



6.3.4 Per Port Platform Customization

The Intel Embedded Graphics Drivers provide what is considered the most useful tools to the embedded market — per port platform customizations. This includes the following:

- Defining custom DTD panel timings

  : `PixelClock, HorzActive, HorzSync` etc...

- Customized GPIO pin selection for I^2C and DDC communication with DVO encoders and panels.

  :`I2cPin, I2cDab, I2cSpeed` etc...

- Flat Panel width and height limitations and power and/or backlight control mechanisms

  :`BkltMethod, BkltT1, BkltT2, GpioPinVdd` etc...

- Port driver specific attribute settings for initialization at boot time.

  : `Brightness, Contrast, H-Position` etc...

All of the above can be set for each individual port depending on the maximum number of ports the chipset supports. Also, you can have multiple instances of these configurations to allow different settings per configuration.

The usage model for this per-config, per-port platform customizations follows after the same options available in the INF registry settings for the Intel Embedded Graphics Drivers for Microsoft Windows XP. Please see Figure 6.3.7, “Sample iegd.reg File” on page 93 or to the provided registry sample file in the IEGD Windows CE driver package for examples. The following sections provide information on these configurations.

6.3.4.1 Per Port Customization – General Port Configuration

This section describes port-specific general configuration options. These options are located under
`[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\Port\1\General]`

- Edid

  This boolean key enables (set to 1) or disables (set to 0) the EdidAvail and EdidNotAvail keys.

- EdidAvail and EdidNotAvail

  These two 16-bit keys control the available timings for the display. If an EDID is successfully read from the display device, then IEGD uses the EdidAvail flag to determine what timings are available. Otherwise, if an EDID cannot be read, then IEGD uses the EdidNotAvail key.

Bit #	Value (0 or 1)
0	Disable/Enable driver built-in timings
1	Disable/Enable EDID timings. (Only valid for the EdidAvail flag)
2	Disable/Enable DTD
3-15	Reserved

- CenterOff
  If the selected frame buffer size is smaller than what the IEGD hardware can support, by default the frame buffer will be centered with a black border around it. To explicitly turn off this feature, the user may set the “CenterOff” key to “1”.
- Rotation and Flip
  IEGD supports desktop rotation through the “Rotation” key in Single, Twin, and Clone mode. Rotation is not supported in Vertical Extended Mode.
  The “Rotation” key can be set to one of the four follow values.

Degree	Key Value
0	0 (default)
90	5A
180	B4
270	10E

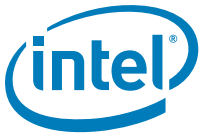
  So, “Rotation”=dword:5A will rotate the frame buffer 90 degrees.
  The “Flip” key flips the desktop horizontally, displaying a mirror image. “Flip” is a boolean value: 1 to enable, 0 to disable.
- Scale
  IEGD can scale the desktop to the output panel using the panel’s DTD or EDID (in that order). “Scale” is a boolean value: 1 to enable, 0 to disable.

6.3.4.2 Per Port Customization – Custom DTD Timings

For each configuration, each port can be added with up to 255 customized DTD modes.

The following is an example of adding 800x640 mode to the LVDS port when ConfigId=1 is used.

[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\Port\4\DTD\1]
“PixelClock”=dword:9c40
“HorzActive”=dword:320
“HorzSync”=dword:28
“HorzSyncPulse”=dword:80
“HorzBorder”=dword:0
“HorzBlank”=dword:100
“HorzSize”=dword:0
“VertActive”=dword:280
“VertSync”=dword:1
“VertSyncPulse”=dword:4
“VertBorder”=dword:0
“VertBlank”=dword:1c
“VertSize”=dword:0
“Flags”=dword:1e



6.3.4.3 Per Port Customization — Custom DVO GPIO Pin Settings

For each configuration, each port's GPIO pin pair settings can be configured in terms of which physical pins, what I^2C slave address the DVO encoder on that port responds to and what speeds to use.

```
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\Port\1\DVO]
      ;  "I2cPin"=dword:2
      ;  "I2cDab"=dword:70
      ;  "I2cSpeed"=dword:0
      ;  "DdcPin"=dword:0
      ;  "DdcSpeed"=dword:0
```

Note: For Per-Config, Per-Port configuration, the subkey path includes the correct `Config` and `Port` numbers.

6.3.4.4 Per Port Customization — Custom Flat Panel Controls

Similarly, the flat panel native resolution and power and backlight sequencing controls can also be configured here.

```
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\Port\1\FPInfo]
      ;  "BkltMethod"=dword:0
      ;  "BkltT1"=dword:0
      ;  "BkltT2"=dword:0
      ;  "BkltT3"=dword:0
      ;  "BkltT4"=dword:0
      ;  "BkltT5"=dword:0
      ;  "GpioPinVdd"=dword:0
      ;  "GpioPinVee"=dword:0
      ;  "GpioPinBklt"=dword:0
      ;  "BkltEnable"=dword:0
      ;  "UseGMCHClockPin"=dword:0
      ;  "UseGMCHDataPin"=dword:0
```

Note: For Per-Config, Per-Port configuration, the subkey path includes the correct "Config" and "Port" numbers

6.3.4.5 Per Port Customization — Attribute Initialization

Attributes are also per config and per port. However, the actual keys are dependant on the port driver being used. Below are examples of registry keys associated with initializing attributes for the Chrontel Port Driver.

Attribute Ranges for the CH7009 Port Driver:

```
      Brightness                0-100
      Contrast                  0-7
      Flicker Filter            0-4
      Saturation                0-7
      Hue                       0-100
      Text Filter               0-3
      Macrovision               boolean
      Overscan ratio            1-4         (Low, Std, High, None)
      TV Format                 1, 17, 33   (NTSC-M, NTSC-M-J,
                                              NTSC-4.33)
      TV Output                 1-4         (Comp & Svid, Comp, Svid,
                                Comp)
```

For complete information on port driver attributes, refer to Appendix B.

Note: For Per-Config, Per-Port configuration, the subkey path includes the correct "Config" and "Port" numbers.

The following example sets the CH7009 port driver attributes using the attribute IDs. Please see Section B.2.4, “Chrontel CH7009/CH7010 Port Driver TV Attributes” on page 165 for a list of CH7009 attribute IDs and their meanings.

```
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\Port\1\Attr]
       “0”=dword:32
       “1”=dword:4
       “3”=dword:1
       “8”=dword:1
       “12”=dword:0
       “14”=dword:1
       “19”=dword:1
```

6.3.5 Miscellaneous Configuration Options

This section covers registry settings not in [HKEY_LOCAL_MACHINE\Drivers\Display\Intel].

6.3.5.1 Text Anti-Aliasing

The Microsoft Windows CE driver supports text anti-aliasing. To switch it on, add these registry settings:

```
[HKEY_LOCAL_MACHINE\System\GDI\Fontsmoothing]
[HKEY_LOCAL_MACHINE\System\GDI]
       "ForceGRAY16"=dword:1
```

Note: Text Anti-Aliasing should always be turned on when using a TV display device.

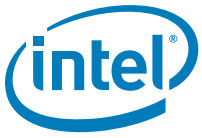
6.3.6 D3D Mobile Support

IEGD 8.0 supports Direct3D* Mobile on Windows CE 5.0 and 6.0. Users need to ensure that their Windows CE target machine platform workspace has been included with D3D Mobile support. This is done by simply dragging the D3D Mobile component from the catalogue view to the workspace in Windows CE 5.0 or turning it on via the checkbox in Windows CE 6.0.

Also, the new IEGD D3D mobile driver binary, “iegd3dg3.dll” (part of the IEGD driver release for Windows CE 5.0 and 6.0) needs to be included in the workspace image. Add this binary into the “.BIB” configuration file of the target platform workspace (see Figure 14):

```
iegd3dg3.dll <specify_path_here>\iegd3dg3.dll NK
```

No other IEGD registry configuration is necessary for this feature to work.



Note: The IEGD Windows CE D3D Mobile feature requires more memory at runtime:

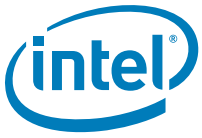
- For Windows CE 5.0, it is recommended that 64 Mbytes is configured via Platform Builder (go to the Platform menu, choose **Settings**, and choose the **Build Options** tab. Ensure the option labeled “Run-time Image Can be Larger than 32 Mbytes (IMGRAM64=1)” is checked).
- For Windows CE 6.0, depending on the number of components built into the target platform workspace, the amount of memory could be significantly more, due in part to the new kernel memory architecture adopted by the OS (assuming all multimedia components are built-in). It is recommended that 84 Mbytes of memory be configured for the target machine workspace image. Refer to the following Microsoft URL for information on how configure more than 64 Mbytes on Windows CE 6.0:

http://msdn2.microsoft.com/en-us/library/aa909457.aspx

6.3.7 Sample iegd.reg File

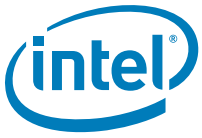
```
;***************** BEGIN INTEL DISPLAY DRIVER REGISTRY ENTRY *****************
;*****************************************************************************
;*---------------------------------------------------------------------------
;* Copyright (c) Intel Corporation (2002 - 2007).
;*
;* The source code contained or described herein and all documents
;* related to the source code ("Material") are owned by Intel
;* Corporation or its suppliers or licensors. Title to the Material
;* remains with Intel Corporation or it suppliers and licensors. The
;* Material contains trade secrets and proprietary and confidential
;* information of Intel or its suppliers and licensors. The Material is
;* protected by worldwide copyright and trade secret laws and
;* treaty provisions. No part of the Material may be used, copied,
;* reproduced, modified, published, uploaded, posted, transmitted,
;* distributed, or disclosed in any way without Intels prior express
;* written permission.
;*
;* No license under any patent, copyright, trade secret or other
;* intellectual property right is granted to or conferred upon you by
;* disclosure or delivery of the Materials, either expressly, by
;* implication, inducement, estoppel or otherwise. Any license
;* under such intellectual property rights must be express
;* and approved by Intel in writing.
;*---------------------------------------------------------------------------
[HKEY_LOCAL_MACHINE\System\GDI\Drivers]
"Display"="ddi_iegd.dll"
;*****************************************************************************
; The Following Sections Provide
; General Driver-Wide Registry Settings
;*****************************************************************************
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel]
;-----------------------------------------------------------------------------
; Following registry entry for
; pcf version used
; 700 : IEGD version
;-----------------------------------------------------------------------------
"PcfVersion"=dword:700
;-----------------------------------------------------------------------------
; This value dictates the configuration to select for Per-Port settings from
; port specific regsitry. The settings mirror Windows XP IEGD drivers
; implementation. Refer to the User’s Guide for details.
;-----------------------------------------------------------------------------
"ConfigId"=dword:1
;-----------------------------------------------------------------------------
; Provide a list of port drivers to attempt to load upon boot time
;-----------------------------------------------------------------------------
```

```
"PortDrivers"="ch7009 ch7017 fs454 lvds ns2501 ns387 sii164 ti410 th164 sdvo tv"
;*****************************************************************************
; The Following Sections Provide Per-Config configuration. The Platform string in
the path can
; be “ALL” for all platforms, or any of the following for platform-specific
configurations:
; Q35, GM965, Q965, 946GZ, 945G, 945GM, 915GV, 915GM, 855GM, and 845G
;*****************************************************************************
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\General]
;-----------------------------------------------------------------------------
; Following registry entries for display settings:resolution, bit depth and
; refresh rate
;
; Width & Height values must be hex, for example
; 1400x1050 : 578h x 41Ah
; 1280x1024 : 500h x 400h
; 1024x768 : 400h x 300h
; 800x600 : 320h x 258h
; 640x480 : 280h x 1E0h
; etc...
;
; In vertical extented mode, height is doubled
; 640x960 : 280h x 3c0
; 800x600 : 320h x 4b0h
; etc...;
;-----------------------------------------------------------------------------
"Width"=dword:320
"Height"=dword:258
;-----------------------------------------------------------------------------
; Bit depth must be one of:
; 8bpp : 8
; 16bpp : 10
; 24bpp : 18
; 32bpp : 20
;-----------------------------------------------------------------------------
"Depth"=dword:10
;-----------------------------------------------------------------------------
; Refresh rate must be in hex:
; 60 : 3c
Intel® Embedded Graphics Drivers and Video BIOS v8.0
September 2007 User’s Guide
Document Number: 274041-013US
Installing and Configuring Microsoft Windows CE Drivers
; 70 : 46
; 75 : 4b
; 85 : 55
; etc...
; any refresh rate as long as the display port supports it refresh of '0' will
; take the first refresh that matches width, height and bpp
;-----------------------------------------------------------------------------
"Refresh"=dword:3c
;-----------------------------------------------------------------------------
; Following is registry entry for controlled configuration of video memory
; usage / location
;
; The following settings are for a 64M platform, where the video memory is 14M
; at the top the above settings are assuming there is no system bios / firmware
; that has stolen memory from top of memory. If it does exist reduce
; ReservedMemorySize avoiding overlap exception for ACSFL, memory area is
; reused
;
; NOTE: CURRENTLY THESE SETTINGS ARE REMARKED FOR DYNAMIC VIDEO MEMORY
; CONFIGURATION
;-----------------------------------------------------------------------------
; "ReservedMemoryBase"=dword:03200000
; "ReservedMemorySize"=dword:00E00000
;-----------------------------------------------------------------------------
```



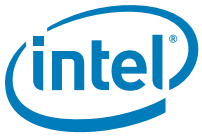
```
; Below is Maximum Frame Buffer Size used to limit the maximum size in bytes
; of the main frame buffer
;-------------------------------------------------------------------------------
"MaxFbSize"=dword:300000
;-------------------------------------------------------------------------------
; Page Request Limit is used to control the max allocations of offscreen video
; surfaces, buffers etc.. value is in number of pages (4K).
; this is independant of dynamic or static memory configuration.
;
; The max for 845, 855, 852 = 128 Mbytes = 0x8000
; The max for 915s, 945s = 256 Mbytes = 0x10000
;-------------------------------------------------------------------------------
"PageReqLimit"=dword:8000
;-------------------------------------------------------------------------------
; Above settings are to define a minimum width and heigh that would allow for
; video surface allocations to succeed, eg: surfaces with width < 16 are
; forced to be in system-mem, surfaces with height < 16 are forced to be in
; system-mem only affects allocations of surfaces with GPE_PREFER_VIDEO_MEMORY
; flag
;-------------------------------------------------------------------------------
"MinVidSurfX"=dword:10
"MinVidSurfY"=dword:10
;-------------------------------------------------------------------------------
; Following are the registry entries for acceleration configuration
;-------------------------------------------------------------------------------
Installing and Configuring Microsoft Windows CE Drivers
Intel® Embedded Graphics Drivers and Video BIOS v8.0
User's Guide September 2007
Document Number: 274041-013US
; Set SysToVidStrech to '1' enables driver to perform System to Video stretch
; blits
;-------------------------------------------------------------------------------
"SysToVidStretch"=dword:1
;-------------------------------------------------------------------------------
; Blend filterting method selection
; The hardware must be capable of
; support, else, emulation is done.
; Possible blend methods are:
; BlendFilter 0 == NEAREST
; BlendFilter 1 == BILINEAR
; BlendFilter 2 == ANISOTROPIC
; BlendFIlter 3 == 4X4
;-------------------------------------------------------------------------------
"BlendFilter"=dword:2
;-------------------------------------------------------------------------------
; Option for enabling/disabling TEARING - Default is OFF
;-------------------------------------------------------------------------------
; Set '1' to enable the NOTEARING option
"TearFB"=dword:1
;-------------------------------------------------------------------------------
; Specify whether to enable d3d
; NO_D3D Value: 0(default)
; : 0 --> Enable D3D
; : 1 --> Disable D3D
;-------------------------------------------------------------------------------
"NO_D3D"=dword:0
;-------------------------------------------------------------------------------
; Select Display configuration, single, twin ...
; Possible Display Config combo:
; DisplayConfig 1 == SINGLE
; (Single is default if none specified)
; DisplayConfig 4 == TWIN
; --> (Twin mode: common timing across ports)
; DisplayConfig 2 == CLONE
; --> (Clone mode: distinct timing per port)
; (845 doesn't support Clone)
; DisplayConfig 5 == VEXT (vertical extend)
```

```
; --> (Vert Extended modes : "Height" )
; ( registry key value must be 2X the )
; ( intended port timings. Both ports )
; ( must use the same timings. For )
; ( example, for port timings of )
; ( 800x600, the DisplayConfig should )
; ( be 5 and the Height=1200 or 0x4b0 )
; ( Overlay wont work in VEXT mode. )
; (845 & 915GV doesn't support Vext)
;-------------------------------------------------------------------------------
"DisplayConfig"=dword:1
;-------------------------------------------------------------------------------
; Select Port Order
; PortOrder specifies the actual port that will be used for the primary and
; secondary ports. IF specified port is unavailable (port driver failed or
; display detection failed or port is not available on current chipset), then
; the next port in the above order will be used. PortOrder must be set,
; based on chipset specifications:
; On i915 chipsets:
; ================================
; 1 - Integrated TV Encoder
; 2 - DVO B port/RGBA port
; 3 - DVO C port
; 4 - Internal LVDS port
; 5 - Analog port
;
; On i830/835/845/85x/865 chipsets:
; ================================
; 1 - DVO A port
; 2 - DVO B port/RGBA port
; 3 - DVO C port
; 4 - Internal LVDS port
; 5 - Analog port
;
; On 835: If RGBA is used (DVO B & C together), then use DVO B number
; to specify any parameter for it.
;
; On i81x chipsets:
; ================
; Port numbers:
; 3 - DVO port
; 5 - Analog port
;-------------------------------------------------------------------------------
"PortOrder"="5432"
;-------------------------------------------------------------------------------
; Set Clone Port resolutions
;-------------------------------------------------------------------------------
; "CloneWidth"=dword:320
; "CloneHeight"=dword:258
; "CloneRefresh"=dword:3c
;-------------------------------------------------------------------------------
; Set "1" to enable Display Detection
; DisplayDetect is to detect display child device before using it
; (panel/tv/etc...).BEWARE, setting this to '1' will mean display for the
; requested port wont be enabled if detection failed. Use this option wisely.
;-------------------------------------------------------------------------------
"DisplayDetect"=dword:0
;-------------------------------------------------------------------------------
; Overlay Color Correction Settings
; Gamma: 32-bit integer in 24i.8f format, ranging from 0.6 - 6.0 decimal
; Brightness: 32-bit integer ranging from 0 to 0xFFFF. 0x8000 = no correction
; Contrast: 32-bit integer ranging from 0 to 0xFFFF. 0x8000 = no correction
; Saturation: 32-bit integer ranging from 0 to 0xFFFF. 0x8000 = no correction
;-------------------------------------------------------------------------------
; "OverlayGammaCorrectR"=dword:100
; "OverlayGammaCorrectG"=dword:100
; "OverlayGammaCorrectB"=dword:100
```



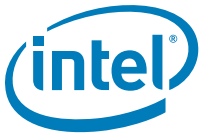
```
; "OverlayBrightnessCorrect"=dword:8000
; "OverlayContrastCorrect"=dword:8000
; "OverlaySaturationCorrect"=dword:8000
;*****************************************************************************
;
; The sections below are for the more detailed per port
; registry configurations. It follows the same usage model and
; key value meanings as the Windows INF registry configuration
; file. Refer to the User's Guide for details.
;*****************************************************************************
;-----------------------------------------------------------------------------
; Config 1 - DVO-B Port (For Almador) |
;-----------------------------------------------------------------------------
; Following are the registry
; entries for port's general config
;-----------------------------------------------------------------------------
;
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\General]
;
; Advanced Edid Configuration
; ---------------------------
; "Edid"=dword:1
; "EdidAvail"=dword:7 ; STD TIMINGS + EDID TIMINGS + USER TIMINGS
; "EdidNotAvail"=dword:7 ; STD TIMINGS + USER TIMINGS
;
; EdidAvail and EdidNotAvail: <only 16 bits used>
; ---------- -------------
; These 2 parameters can be used to control the available timings for
; any display. 'EdidAvail' is used when EDID is read from the display
; device. If an attempt to read EDID is failed or 'Edid = 0' then
; driver uses 'EdidNotAvail' flags.
;
; See below bit definitions for both 'EdidAvail' and 'EdidNotAvail'
;
; BIT 0:
; ------
; 0 - Do not use driver built-in standard timings
; 1 - Use driver built-in standard timings
;
; BIT1: <not applicable to EdidNotAvail>
; ------
; 0 - Do not use EDID block
; 1 - Use EDID block and filter modes
;
; BIT2:
; ------
; 0 - Do not use user-DTDs
; 1 - Use user-DTDs.
;
; BIT3-BIT15
; -----------
; Future use.
;
; Default behavior:
; -----------------
; If user doesn't provide EdidAvail and EdidNotAvail, then
; EdidAvail = Use Std timings + Use EDID block and Filter modes
; EdidNotAvail = Use Std timings
;
; Rotation Configuration
; ----------------------
; "Rotation"=dword:0
; Rotation entries must be at a right
; angle. An invalid entry will be ignored and
; and no rotation will happen for primary.
; In clone or twin modes, the secondary
; port defaults to follow the primary (if set)
; 0 degrees = 0 (not rotated = default)
```

```
; 90 degrees = 5A
; 180 degrees = B4
; 270 degrees = 10E
;
; Flip Configuration
; ----------------------
; "Flip"=dword:0
; Flip has a valid entry of 1 to turn on
; and 0 to turn off. When turn on the display
; will be horizontally flip.
;
; Rendered Scaling Configuration
; ------------------------------
; "Scale"=dword:0
; Scale works as a boolean switch. Valid
; entries are zero or 1. When "Scale" = 1,
; IEGD will scale the requested framebuffer
; resolution to the fixed native panel size
; indicated by per-port FPInfo, User-DTD or
; EDID (in that order).
; In clone or twin modes, the secondary
; port defaults to follow the primary (if set)
;-------------------------------------------------------------------------
; Following are the registry entries
; for port's DVO I2C settings
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\DVO]
; "I2cPin"=dword:2
; "I2cDab"=dword:70
; "I2cSpeed"=dword:0
; "DdcPin"=dword:0
; "DdcSpeed"=dword:0
;-------------------------------------------------------------------------
; Following are the registry entries
; for port's flat panel's mode-limits,
; power and backlight control
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\FPInfo]
; Only need Width & Height if Panel cannot accept other timings
; "BkltMethod"=dword:3
; "BkltT1"=dword:1E
; "BkltT2"=dword:4
; "BkltT3"=dword:4
; "BkltT4"=dword:14
; "BkltT5"=dword:1F4
; "GpioPinVdd"=dword:27
; "GpioPinVee"=dword:26
; "GpioPinBklt"=dword:28
; "UseGMCHClockPin"=dword:0
; "UseGMCHDataPin"=dword:0
;-------------------------------------------------------------------------
; Following are the registry entries
; for ports first custom DTD mode to add
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\DTD\1]
; "PixelClock"=dword:9c40
; "HorzActive"=dword:320
; "HorzSync"=dword:28
; "HorzSyncPulse"=dword:80
; "HorzBorder"=dword:0
; "HorzBlank"=dword:100
; "HorzSize"=dword:0
; "VertActive"=dword:280
; "VertSync"=dword:1
; "VertSyncPulse"=dword:4
; "VertBorder"=dword:0
; "VertBlank"=dword:1c
```



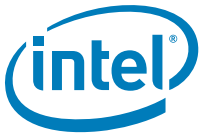
```
; "VertSize"=dword:0
; "Flags"=dword:1e
;-------------------------------------------------------------------------
; Following are the registry entries
; for ports second custom DTD mode to add
; (Up to 255 can be added)
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\DTD\2]
; "PixelClock"=dword:9c40
; "HorzActive"=dword:320
; "HorzSync"=dword:28
; "HorzSyncPulse"=dword:80
; "HorzBorder"=dword:0
; "HorzBlank"=dword:100
; "HorzSize"=dword:0
; "VertActive"=dword:258
; "VertSync"=dword:1
; "VertSyncPulse"=dword:4
; "VertBorder"=dword:0
; "VertBlank"=dword:1c
; "VertSize"=dword:0
; "Flags"=dword:1e
;-------------------------------------------------------------------------
; Following are the registry
; entries for the port device'
; display attribute parameters
; Use when enabling Port device
; example below is for Conexant
; on Port2 (DVO-B for almador)
; key names depend on port driver
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\Attr]
; "Brightness"=dword:32
; "Contrast"=dword:4
; "Flicker Filter"=dword:1
; "Saturation"=dword:4
; "Hue"=dword:32
; "Text Filter"=dword:0
; "Macrovision"=dword:0
; "Overscan ratio"=dword:1
; "TV Format"=dword:1
; "TV Output"=dword:1
; "Composite and S-Video"=dword:1
;-------------------------------------------------------------------------
; Config 1 - Analog Port (For Any Chipset)
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\5\General]
; "Edid"=dword:1
; "EdidAvail"=dword:7 ; STD TIMINGS + EDID TIMINGS + USER TIMINGS
; "EdidNotAvail"=dword:7 ; STD TIMINGS + USER TIMINGS
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\Config\1\Port\5\attr]
; GAMMA, BRIGHTNESS, CONTRAST
; "35"=dword:a0a0a0 ; gamma: 3i.5f format for R-G-B, ranging 0.6 to 6
; "36"=dword:808080 ; brightness: 0 to FF, 0x80 is no correction
; "37"=dword:808080 ; contrast: 0 to FF, 0x80 is no correction
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\Config\1\Port\5\DTD\1]
; "PixelClock"=dword:9c40
; "HorzActive"=dword:320
; "HorzSync"=dword:28
; "HorzSyncPulse"=dword:80
; "HorzBorder"=dword:0
; "HorzBlank"=dword:100
; "HorzSize"=dword:0
; "VertActive"=dword:280
; "VertSync"=dword:1
; "VertSyncPulse"=dword:4
; "VertBorder"=dword:0
```

```
; "VertBlank"=dword:1c
; "VertSize"=dword:0
; "Flags"=dword:1e
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\5\DTD\2]
; "PixelClock"=dword:9c40
; "HorzActive"=dword:320
; "HorzSync"=dword:28
; "HorzSyncPulse"=dword:80
; "HorzBorder"=dword:0
; "HorzBlank"=dword:100
; "HorzSize"=dword:0
; "VertActive"=dword:258
; "VertSync"=dword:1
; "VertSyncPulse"=dword:4
; "VertBorder"=dword:0
; "VertBlank"=dword:1c
; "VertSize"=dword:0
; "Flags"=dword:1e
;-------------------------------------------------------------------------
; Config 1 - Int-LVDS Port (For 855 or 915GM) |
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\4\General]
; "Edid"=dword:1
; "EdidAvail"=dword:7 ; STD TIMINGS + EDID TIMINGS + USER TIMINGS
; "EdidNotAvail"=dword:7 ; STD TIMINGS + USER TIMINGS
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\4\FPInfo]
; Only need Width & Height if Panel cannot except other timings
; "BkltMethod"=dword:0
; "BkltT1"=dword:0
; "BkltT2"=dword:0
; "BkltT3"=dword:0
; "BkltT4"=dword:0
; "BkltT5"=dword:0
; "GpioPinVdd"=dword:0
; "GpioPinVee"=dword:0
; "GpioPinBklt"=dword:0
; "UseGMCHClockPin"=dword:0
; "UseGMCHDataPin"=dword:0
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\4\Attr]
; "27"=dword:1
; Attribute "27" = Dual Channel (boolean)
; "18"=dword:1
; Attribute "18" = Panel Fit Upscale (boolean)
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\4\DTD\1]
; "PixelClock"=dword:9c40
; "HorzActive"=dword:320
; "HorzSync"=dword:28
; "HorzSyncPulse"=dword:80
; "HorzBorder"=dword:0
; "HorzBlank"=dword:100
; "HorzSize"=dword:0
; "VertActive"=dword:280
; "VertSync"=dword:1
; "VertSyncPulse"=dword:4
; "VertBorder"=dword:0
; "VertBlank"=dword:1c
; "VertSize"=dword:0
; "Flags"=dword:1e
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\4\DTD\2]
; "PixelClock"=dword:9c40
; "HorzActive"=dword:320
; "HorzSync"=dword:28
; "HorzSyncPulse"=dword:80
; "HorzBorder"=dword:0
; "HorzBlank"=dword:100
; "HorzSize"=dword:0
; "VertActive"=dword:258
```



```
; "VertSync"=dword:1
; "VertSyncPulse"=dword:4
; "VertBorder"=dword:0
; "VertBlank"=dword:1c
; "VertSize"=dword:0
; "Flags"=dword:1e
;-------------------------------------------------------------------------
; Config 1 - SDVO Port-B (For Napa)
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1]
; "name"="IEGD SDVO Configuration File"
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2]
; "name"="svga"
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\FPInfo]
; For a SDVO driver, sample settings for the panel:1400x1050
; Only need Width & Height if Panel cannot except other timings
; "Width"=dword:578
; "Height"=dword:41A
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\Attr]
; "27"=dword:1
; Attribute "27" = Dual Channel (boolean)
; Optional - Only enable for font anti-aliasing
; Enabling this causes minor performance impact
; Only recommended for TV Output.
;[HKEY_LOCAL_MACHINE\System\GDI\Fontsmoothing]
;
;[HKEY_LOCAL_MACHINE\System\GDI]
; "ForceGRAY16"=dword:1
;****************** INTEL DISPLAY DRIVER REGISTRY ENTRY END *****************
```

This page is intentionally left blank.



7.0 Installing and Configuring Linux Drivers

This chapter describes the configuration and installation of the IEGD for Linux systems. The IEGD supports X-servers from the X.org* and XFree86* organizations.

The Intel Linux driver is for use with the integrated graphics of Intel chipsets on the Embedded Intel Architecture roadmap. The driver supports 8-, 16- and 24-bit pixel depths, dual independent head configuration on capable hardware, flat panel, hardware 2D acceleration, hardware cursor, the XV extension, and the Xinerama extension.

7.1 Overview

Kernel patches, separate DRM modules, and kernel recompilation were all necessary in previous versions of the Intel Embedded Graphics Driver. In version 8.0, the IEGD Kernel Module (IKM) contains a combination of the agpgart and DRM modules which must be present for the Intel Embedded Graphic Driver. Both modules have been modified for the IEGD architecture and are combined with the Linux kernel.

The IEGD Linux distribution package contains drivers built for the following X-Servers:

- XFree86 version 4.3.0
- X.org version 6.7
- X.org version 6.8
- X.org version 7.0
  - — Used for X.org 6.9 or 7.0
- X.org version 7.1
- X.org version 7.2

The IEGD has been tested with the official version of these servers from the www.Xfree86.org or the www.X.org Web sites and may not operate with other versions of these servers.

7.2 Prerequisites

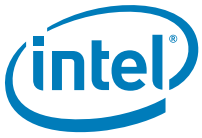
The following lists the prerequisites for installing and configuring the IEGD Linux driver.

- Platform with supported Intel chipset.
- Platform with a minimum of 128 Mbytes.
- Resolution and timing specifications for the display devices that will be configured.
- Driver package consisting of directories and files (see the following reduced samples, which are located under the `IEGD Linux` directory).

Note: In the following, “Xorg-X11R7.0” is an example X server version that should be replaced with the version to be used.

- Documents/Relnotes
  Documents/UsersGuide.pdf
  Documents/Xorg-X11R7.0/iegd.4
  Documents/Xorg-X11R7.0/IntelEscape.3x
  Documents/Xorg-X11R7.0/IntelEscape.3x
- License/License.txt
- Driver/<xserver name>/iegd_drv.o (or iegd_drv.so for Xorg 7.0)
  Driver/<xserver name>/libXlibXiegd_escape.a
  Driver/<xserver name>/libXiegd_escape.so.2.0.0
  Driver/<xserver name>/iegd_escape.h
  Driver/<xserver name>/ch7009.so
  Driver/<xserver name>/ch7017.so
  Driver/<xserver name>/fs454.so
  Driver/<xserver name>/lvds.so
  Driver/<xserver name>/tv.so
  Driver/<xserver name>/ns2501.so

  Patch files are located in the IEGD_Patches directory. The following are sample patch file names. If you do not see a GART patch that matches the Linux product you are using, see Section 7.4, “IKM Patch Instructions” on page 114. You only need to use one of the patching methods, either the IKM or the GART patch.
  - agpgart.patch-2.4.20 (Patch for 2.4.20 kernel)
  - agpgart.patch-2.4.20-8 (Patch for Red Hat 9 2.4.20-8 kernel)
  - agpgart.patch-2.4.24 (Patch for 2.4.24 kernel)
  - agpgart.patch-2.4.26 (Patch for 2.4.26 kernel)
  - agpgart.patch-2.6.5-1.358 (Patch for Fedora Core 2 2.6.5-1.358 kernel)
  - agpgart.patch-2.6.5-7.191 (Patch for older SUSE 2.6.5-7.191 kernel)
  - agpgart.patch-2.6.5-7.244 (Patch for NLPOS 9 SP3 2.6.5-7.244 kernel)
  - agpgart.patch-2.6.13-15 (Patch for SUSE 10 2.6.13-15 kernel)
  - agpgart.patch-2.6.15-1.2054 (Patch for Fedora Core 5 2.6.15-1.2054 kernel)
  - agpgart.patch-2.6.16.46-0.12 (Patch for SUSE 10 SP1 2.6.16.46-0.12 kernel)
  - agpgart.patch-2.6.18-1.2798 (Patch for Fedora 6 2.6.18-1.2798 kernel)
  - agpgart.patch-2.6.21-1.3194 (Patch for Fedora 7 2.6.21-1.3194 kernel)
  - COPYING (GPL* license agreement for patches)
- Linux kernel source tree for active running kernel.
  - AGPGART must be enabled as a module.
  - Direct Rendering support enabled.
- XFree86 version 4.2, XFree86 version 4.3, X.org version 6.7.0, X.org version 6.8.2, or X.org version 7.0 X-Server installed and functional.
  - Directory paths to XFree86 or X.org installation and configuration files.



- Other system capabilities
  - Ncurses (SUSE 10), etc.
  - IEGD Kernel Module for GART and DRM patches
- System administration privileges.

7.2.1 Supported Hardware

IEGD supports the following chipsets with integrated graphics:

- Intel® Q35 Express chipset
- Mobile Intel® GME965 Express chipset
- Intel® Q965 Express chipset
- Mobile Intel® 945GME Express chipset
- Intel® 945G Express chipset
- Intel® 915GV Express chipset
- Mobile Intel® 915GME Express chipset
- Mobile Intel® 910GMLE Express chipset
- Intel® 855GME chipset
- Intel® 852GME chipset
- Intel® 852GM chipset

7.3 Installation

To install the IEGD, you must first obtain the `IEGDLinux.tgz` file from the following Web address:

http://www.intel.com/go/IEGD

From the “Related information” box, select **Download Drivers**, and then in “select your O/S” select **Linux**. Select the latest IEGD x.x with CED and follow the instructions for downloading the file. Please see the section on the Configuration Editor (CED) for instructions on configuring the driver.

After you have the `IEGDLinux.tgz` file, you can install the IEGD by performing the instructions in the following section(s). Please see the instructions for your specific distribution. If you are using a Linux distribution different from those for which these instructions are designed, you may need to adapt the steps for your specific situation.

7.3.1 Fedora Core 5

1. Log into to a system administration account.
2. Untar the driver package to a convenient location.

   tar -xvzf <driver package.tgz>

   This creates a directory structure in the directory where you extracted the .tgz file and contains the following directories:

   — IEGD_X_x_Linux - Contains the Documents, Driver, License, and Utilities subdirectories, where X is the major version and x is the minor version.

   The Documents subdirectory contains the Xfree86 and Xorg-X11 subdirectory. These directories contain man pages for the IEGD.

   The Driver directory contains subdirectories the supported versions of XFree and Xorg.

   The Utilities directory contains IEGD utilities, including the `iegdgui` runtime configuration utility.

   — IEGD_Patches - Contains files for patching the Linux kernel agpgart module.
3. Check the version of the X-Server your system is running. Type the following command:

   X -version

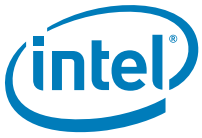
Note: FC5 is normally Xorg 11 7.0.0.

4. Copy the IEGD driver binary, `iegd_drv.o` (or `iegd_drv.so`), from the `IEGD_X_x_Linux/Driver/<xserver name>` directory to the X-Server's modules/drivers directory, where X is the major version and x is the minor version. For FC5 (X.org-7.0 based distribution), the default location is `/usr/lib/X11/lib/modules/drivers`. This location can vary by distribution so check your system for the proper path and replace X and x with the correct version numerals.

   cd IEGD_8_0_Linux/Driver/Xorg-X11R7.0
   cp iegd_drv.so /usr/lib/xorg/modules/drivers
5. Copy the necessary port driver files (*.so files in the `IEGD_X_x_Linux/Driver/<xserver name>` directory) to the X-Server lib/modules directory, where X is the major version and x is the minor version. The default installation location is `/usr/lib/xorg/modules`. This location can vary, so check your system for the proper path. After the required port drivers have been copied, you can specify them in the PortDrivers option in the Device section of the config file. For more information on specifying the PortDrivers option, refer to Table 23, “Supported Driver Options” on page 122. For example, to copy all the port drivers use the following command:

   cp *.so /usr/lib/xorg/modules
6. Copy the escape control library `libXiegd_escape.so.2.0.0` from the `IEGD_X_x_Linux/Driver/<xserver name>` directory to the X-Server library directory, where X is the major version and x is the minor version. The default installation location is `/usr/lib`. For example,

   cp libXiegd_escape.so.2.0.0 /usr/lib



7. In the X-Server library directory, create symbolic links for the escape library aliases:

   cd /usr/lib
   ln -s libXiegd_escape.so.2.0.0 libXiegd_escape.so
   ln -s libXiegd_escape.so.2.0.0 libXiegd_escape.so.2
   ldconfig

8. From the `IEGD_X_x_Linux/Documents` directory, where X is the major version and x is the minor version, copy the driver man page, `iegd.4`, to the `man/man4` directory. The default installation location is `/usr/X11R6/man/man4`. This location can vary by distribution so check your system for the proper path and replace X and x with the correct version numerals. For example:

   cd IEGD_8_0_Linux/Documents/Xorg-X11
   cp iegd.4 /usr/X11R6/man/man4

9. Patch the agpgart module with the Intel extensions. **Skip this step if you are using the IKM method detailed in** Section 7.4**.**

Note: You must have your kernel source and GCC installed for this step. If you do not have these installed, you will need to follow the kernel install instructions on the Redhat Web site and use your software installer for development tools. At this point you should also ensure that the kernel is set for loadable modules.

You can use the IKM method described in Section 7.4, “IKM Patch Instructions” on page 114 instead of using the instructions below. You only need to use one of the patching methods, either the IKM or the GART patch.

To patch a Linux 2.6.xxxx kernel with the GART changes:

a. cd into the kernel source directory (e.g., `cd /usr/src/linux-2.x.xxxx`). For FC5 it is:

   cd /usr/src/redhat/BUILD/kernel-2.6.15/linux-2.6.15.i686

b. Execute the patch command (e.g., `patch -p1 < .../IEGD_Patches/Driver/agpgart.patch-2.x.xxxx`). For FC5 use:

   patch -p1 < .../IEGD_Patches/Driver/agpgart.patch-2.6.15-1.2054

To update the kernel:

a. cd to the kernel source directory (see above)

b. Since the agpgart is typically built-in by default, it needs to be configured to install it as a module before updating the kernel.

   make menuconfig

   From the configuration menu, select `Device Drivers`, then `Character Devices`. Scroll down to `/dev/agpgart`. Change the disposition to `M` (for Module). Exit from the configuration menu and save your changes.

c. Execute the make all modules command:

   make all

d. Install the modules and build the kernel:

   make modules_install && make install

e. Reboot.

f. If you did not edit the `grub.conf` file, select the custom kernel.

Note: Make sure that the newly updated kernel is the one being booted.

g. Run the following commands:

modprobe agpgart
modprobe intel-agp

h. To ensure the agpgart modules are loaded after a reboot, add the following line to the `/etc/modprobe.conf` file:

alias char-major-10-175 intel-agp

i. Modify your `xorg.conf` file to include a device section for this driver and a Monitor section for your display. See Section 7.6 for details on the driver configuration and the list of supported options. The default installation location for this file is `/etc/X11`

Note: You may need to disable SELinux security to allow the IEGD driver to load. Alternatively, you will need to configure SELinux to allow IEGD to operate.

j. Reboot.

At this point, when X Windows starts, it should be using the IEGD driver. To verify this, you can check your Xorg log, or run the `iegdgui` utility (found in the Utilities directory).

Note: You may need to set the `iegdgui` file properties to be executable before it will run.

7.3.2 SUSE 10

1. Log into to a system administration account.
2. Untar the driver package to a convenient location.

   tar -xvzf <driver package.tgz>

   This creates a directory structure in the directory where you extracted the `.tgz` file and contains the following directories:

   — `IEGD_X_x_Linux` - Contains the Documents, Driver, License, and Utilities subdirectories, where X is the major version and x is the minor version.

   The Documents subdirectory contains the Xfree86 and Xorg-X11 subdirectory. These directories contain man pages for the IEGD.

   The Driver directory contains subdirectories for the supported versions of XFree and Xorg.

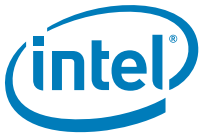
   The Utilities directory contains IEGD utilities, including the `iegdgui` runtime configuration utility.

   — `IEGD_Patches` - Contains files for patching the Linux kernel agpgart module.
3. Check the version of the X-Server your system is running. Type the following command:

   X -version

Note: SUSE 10 is normally Xorg 11 6.8.2.

4. Copy the IEGD driver binary, `iegd_drv.o` (or `iegd_drv.so`), from the `IEGD_X_x_Linux/Driver/<xserver name>` directory, where X is the major version and x is the minor version, to the X-Server's `modules/drivers` directory. For SUSE 10 (X.org-6.8 based distribution), the default location is `/usr/lib/X11/lib/modules/drivers`.



This location can vary by distribution so check your system for the proper path and replace X and x with the correct version numerals.

```
cd IEGD_8_0_Linux/Driver/Xorg-X11R6.8
cp iegd_drv.o /usr/X11R6/lib/modules/drivers
```

Note: Depending on your installation you may need to copy the IEGD driver to an alternate location:

```
cp iegd_drv.o /usr/lib/xorg/modules/drivers
```

5. Copy the necessary port driver files (*.so files in the `IEGD_X_x_Linux/Driver/<xserver name>` directory) to the X-Server `lib/modules` directory, where X is the major version and x is the minor version. The default installation location is `/usr/X11R6/lib/modules`. This location can vary, so check your system for the proper path. Once the required port drivers have been copied, you can specify them in the `PortDrivers` option in the Device section of the config file. For more information on specifying the PortDrivers option, refer to Table 23, “Supported Driver Options” on page 122. For example, to copy all the port drivers use the following command:

   ```
   cp *.so /usr/X11R6/lib/modules
   ```

6. Copy the escape control library `libXiegd_escape.so.2.0.0` from the `IEGD_X_x_Linux/Driver/<xserver name>` directory to the X-Server library directory, where X is the major version and x is the minor version. The default installation location is `/usr/X11R6/lib`. For example,

   ```
   cp libXiegd_escape.so.2.0.0 /usr/X11R6/lib
   ```

7. In the X-Server library directory, create symbolic links for the escape library aliases:

   ```
   cd /usr/X11R6/lib
   ln -s libXiegd_escape.so.2.0.0 libXiegd_escape.so
   ln -s libXiegd_escape.so.2.0.0 libXiegd_escape.so.2
   ldconfig
   ```

8. From the `IEGD_X_x_Linux/Documents` directory, copy the driver man page, `iegd.4`, to the `man/man4` directory, where X is the major version and x is the minor version. The default installation location is `/usr/X11R6/man/man4`. This location can vary by distribution so check your system for the proper path and replace X and x with the correct version numerals. For example:

   ```
   cd IEGD_8_0_Linux/Documents/Xorg-X11
   cp iegd.4 /usr/X11R6/man/man4
   ```

9. Patch the agpgart module with the Intel extensions.

Note: You must have your kernel source and GCC installed for this step. If you do not have these installed, you can use the YAST and search for “kernel”, select it, then search for “GCC” and select it. At this point you should also ensure that the kernel is set for loadable modules.

You can use the IKM method described in Section 7.4, “IKM Patch Instructions” on page 114 instead of using the instructions below. You only need to use one of the patching methods, either the IKM or the GART patch.

To patch a Linux 2.6.xxxx kernel with the GART changes:

a. cd into the kernel source directory (e.g., `cd /usr/src/linux-2.x.xxxx`). For SUSE 10 it is:

cd /usr/src/linux-2.6.13-15

b. Execute the patch command (e.g., `patch -p1 < .../IEGD_Patches/Driver/agpgart.patch-2.x.xxxx`). For SUSE 10 use:

patch -p1 < .../IEGD_Patches/Driver/agpgart.patch-2.6.13-15

To update the kernel:

a. cd to the kernel source directory (e.g., `cd /usr/src/linux-2.x.xxxx`)

cd /use/src/linux-2.6.13-15

b. Since the agpgart is typically built-in by default, it needs to be configured to install it as a module before updating the kernel.

make menuconfig

From the configuration menu, select `Device Drivers`, then `Character Devices`. Scroll down to `/dev/agpgart`. Change the disposition to M (for Module). Exit from the configuration menu and save your changes.

c. Execute the make all modules command:

make all

d. Install the modules and build the kernel:

make modules_install && make install

e. Reboot.

f. If you did not edit the grub.conf file, select the custom kernel.

Note: Make sure that the newly updated kernel is the one being booted.

g. Run the following commands:

modprobe agpgart
modprobe intel-agp

h. To ensure the agpgart modules are loaded after a reboot, add the following line to the `/etc/modprobe.conf` file:

alias char-major-10-175 intel-agp

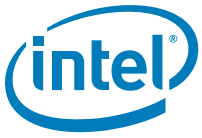
i. Modify your `xorg.conf` file to include a device section for this driver and a Monitor section for your display. See 7.6 for details on the driver configuration and the list of supported options. The default installation location for this file is `/etc/X11`

Note: The CED utility generates generic Linux X config samples using "`Screen0`" and "`Monitor0`" entries but a standard install of SUSE 10 seems to require "`Screen[0]`" and "`Monitor[0]`" in the `Xorg.conf` file.

j. Reboot.

At this point, when X Windows starts, it should be using the IEGD driver. To verify this, you can check your Xorg log, or run the `iegdgui` utility (found in the Utilities directory).

Note: You may need to set the `iegdgui` file properties to be executable before it will run.



7.3.3 Novell Linux POS 9 SP3

1. Log into a system administration account.
2. Untar the driver package to a convenient location.

   tar -xvzf <driver package.tgz>

   This creates a directory structure in the directory where you extracted the `.tgz` file and contains the following directories:

   — `IEGD_X_x_Linux` - Contains the Documents, Driver, License, and Utilities subdirectories, where X is the major version and x is the minor version.

   The Documents subdirectory contains the `Xfree86` and `Xorg-X11` subdirectory. These directories contain man pages for the IEGD.

   The Driver directory contains subdirectories for **`XFree86-4.3`**, `Xorg-6.7`, `Xorg-6.8`, and `Xorg-7.0`.

   The Utilities directory contains IEGD utilities, including the `iegdgui` runtime configuration utility.

   — `IEGD_Patches` - Contains files for patching the Linux kernel agpgart module.

Note: You must have the following installed to successfully install IEGD gcc, patch, and make.

3. Check the version of the X-Server your system is running. Type the following command:

   X -version

Note: NLPOS 9 is normally XFree86 4.4.0 RC 2.

4. Copy the IEGD driver binary, `iegd_drv.o` (or `iegd_drv.so`), from the `IEGD_X_x_Linux/Driver/<xserver name>` directory, where X is the major version and x is the minor version, to the X-Server's `modules/drivers` directory. For NLPOS 9 (XFree86 based distribution), the default location is `/usr/X11R6/lib/modules/drivers/`.

   This location can vary by distribution so check your system for the proper path and replace X and x with the correct version numerals.

   cd IEGD_8_0_Linux/Driver/Xorg-X11R6.8
   cp iegd_drv.o /usr/X11R6/lib/modules/drivers/

5. Copy the necessary port driver files (`*.so` files in the `IEGD_X_x_Linux/Driver/<xserver name>` directory) to the X-Server `lib/modules` directory, where X is the major version and x is the minor version. The default installation location is `/usr/X11R6/lib/modules`. This location can vary, so check your system for the proper path. After the required port drivers have been copied, you can specify them in the PortDrivers option in the Device section of the config file. For more information on specifying the PortDrivers option, refer to Table 23, “Supported Driver Options” on page 122. For example, to copy all the port drivers use the following command:

   cp *.so /usr/X11R6/lib/modules

6. Copy the escape control library `libXiegd_escape.so.2.0.0` from the `IEGD_X_x_Linux/Driver/<xserver name>` directory to the X-Server library directory, where X is the major version and x is the minor version. The default installation location is `/usr/X11R6/lib`. For example,

   cp libXiegd_escape.so.2.0.0 /usr/X11R6/lib

7. In the X-Server library directory, create symbolic links for the escape library aliases:

   cd /usr/X11R6/lib
   ln -s libXiegd_escape.so.2.0.0 libXiegd_escape.so
   ln -s libXiegd_escape.so.2.0.0 libXiegd_escape.so.2
   ldconfig

8. From the `IEGD_X_x_Linux/Documents` directory, copy the driver man page, `iegd.4`, to the `man/man4` directory, where X is the major version and x is the minor version. The default installation location is `/usr/X11R6/man/man4`. This location can vary by distribution so check your system for the proper path and replace X and x with the correct version numerals. For example:

   cd IEGD_8_0_Linux/Documents/Xorg-X11
   cp iegd.4 /usr/X11R6/man/man4

9. Patch the agpgart module with the Intel extensions. **Skip this step if you are using the IKM method detailed in** Section 7.4**.**

Note: You must have your kernel source and GCC installed for this step. If you do not have these installed, you can use the YAST and search for “kernel”, select it, then search for “GCC” and select it. At this point you should also ensure that the kernel is set for loadable modules.

You can use the IKM method described in Section 7.4, “IKM Patch Instructions” on page 114 instead of using the instructions below. You only need to use one of the patching methods, either the IKM or the GART patch.

To patch a Linux 2.6.xxxx kernel with the GART changes:

a. cd into the kernel source directory (e.g., `cd /usr/src/linux-2.x.xxxx`). For NLPOS it is:

   cd /usr/src/linux-2.6.5-7.244

b. Execute the patch command (e.g., `patch -p1 < .../IEGD_Patches/Driver/agpgart.patch-2.x.xxxx`). For NLPOS use:

   patch -p1 < .../IEGD_Patches/Driver/agpgart.patch-2.6.5-7.244

To update the kernel:

a. cd to the kernel source directory (e.g., cd /usr/src/linux-2.x.xxxx)

   cd /use/src/linux-2.6.5-7.244

b. Since the agpgart is typically built-in by default, it needs to be configured to install it as a module before updating the kernel.

   make menuconfig

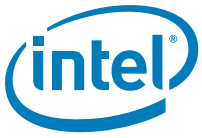
   From the configuration menu, select `Device Drivers`, then `Character Devices`. Scroll down to `/dev/agpgart`. Change the disposition to `M` (for Module). Exit from the configuration menu and save your changes.

c. Execute the make all modules command:

   make all

d. Install the modules and build the kernel:

   make modules_install && make install



e. Reboot.

f. If you did not edit the grub.conf file, select the custom kernel.

Note: Make sure that the newly updated kernel is the one being booted.

g. Run the following commands:

```
modprobe agpgart
modprobe intel-agp
```

h. To ensure the agpgart modules are loaded after a reboot, add the following line to the `/etc/modprobe.conf` file:

```
alias char-major-10-175 intel-agp
```

i. Modify your xorg.conf file to include a device section for this driver and a Monitor section for your display. See Section 7.6, “Configuration” on page 116 for details on the driver configuration and the list of supported options. The default installation location for this file is `/etc/X11`.

Note: The CED utility generates generic Linux X config samples using “`Screen0`” and “`Monitor0`” entries but a standard install of NLPOS seems to require “`Screen[0]`” and “`Monitor[0]`” in the `Xorg.conf` file.

j. Reboot.

At this point, when X Windows starts, it should be using the IEGD driver. To verify this, you can check your Xorg log, or run the `iegdgui` utility (found in the Utilities directory).

Note: You may need to set the `iegdgui` file properties to be executable before it will run.

7.3.4 Other Kernels

The above instructions for FC5 and SUSE 10 describe how to patch and use IEGD with a 2.6.x kernel. If you are using a 2.4.x kernel, the patch step is slightly different:

1. To patch the Linux 2.4.2x kernel with the GART changes:
   a. cd into the kernel source directory (e.g., `cd /usr/src/linux-2.x.xxx`)
   b. Execute the patch command. For example:

```
patch -p1 < .../IEGD_Patches/Driver/agpgart.patch-2.x.xxx
```

2. To update the kernel:
   a. cd to the kernel source directory (e.g., cd /usr/src/linux-2.x.xxx)
   b. Execute the make modules command.

```
make modules
```

   c. Install the modules.

```
make modules_install
```

3. Reboot and if you did not edit the grub.conf file, select the custom kernel.
4. Run the following commands:

```
modprobe agpgart
modprobe intel-agp
```

5. To ensure the agpgart modules are loaded after a reboot, add the following line to the `/etc/modprobe.conf` file:

   alias char-major-10-175 intel-agp

6. Modify your XF86Config or xorg.conf file to include a device section for this driver and a Monitor section for your display. See Section 7.6, "Configuration" for details on the driver configuration and the list of supported options. The default installation location for this file is `/etc/X11`.
7. Reboot.

7.4 IKM Patch Instructions

Note: The IKM process is designed to replace the need to patch your kernel GART and DRM. If you plan to use the patch method of installing IEGD, do NOT use this IKM process. Using the IKM process is the preferred method for installing IEGD.

7.4.1 Finding and Installing the Kernel Source (Headers)

- Building the IKM requires a kernel build tree.
  This should be at least a partial kernel source tree and most importantly, the exact kernel `.config` file used for the build as well as several files generated at kernel build time.
- KERNEL_VERSION is the output of the command **uname -r**
- If you use a kernel from your distribution you will typically have a package with all the files required to build kernel modules for your kernel image.
  - On Fedora and compatibles (e.g., RedHat) this is the `kernel-devel` package. Or if you run `kernel-smp` or `kernel-xen`, you need `kernel-smp-devel` or `kernel-xen-devel`, respectively.
  - On SUSE you need the package kernel-source.
  - In some distributions (e.g., in RHEL or Fedora) the installation of the `kernel-devel` / `kernel-headers` package will be a newer version than the one you currently run. In such a case, you may need to upgrade the kernel package itself and reboot.

7.4.2 Fedora

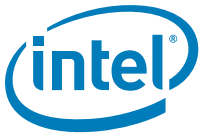
1. Install `kernel-$ARCH-devel`. The version of the package must be the same as the running kernel. Replace `$ARCH` with architecture of the kernel (e.g., smp).
2. Install `Kernel-devel` from the CD/DVD or through the `yum` utility.
3. If you are using a CD or DVD, search for the rpm package for `kernel-$ARCH-devel` and install using

   rpm -ivh kernel-$ARCH-devel

4. For the `yum` utility, type the following command:

   yum install kernel-devel

   This will install `kernel-devel` and resolve dependencies. Note that, as stated earlier, `kernel-devel` that installed through this method might be not the same version as a running kernel. In this case IKM compilation might be successful, however, when trying to insert it to the running kernel, **modprobe/insmod** will produce an error. The solution is to upgrade the kernel-package itself and reboot to make Linux run the updated kernel.



5. Compile the module using the following commands:

   cd IEGD_Patches/IKM
   ./install.sh
   depmod -a
   modprobe iegd_mod

7.4.3 SUSE

1. Install `kernel-source.$ARCH.rpm`.
2. Change to the `/usr/src/linux` directory.
3. Configure the kernel using

   make cloneconfig

4. Create files required for compiling external modules:

   make scripts && make prepare

5. Change to directory `/lib/modules/<runningkernelversion>`
6. Delete 'build' links by running

   rm -rf build

7. Create a build links file pointing to the kernel source:

   ln -s /usr/src/linux-<kernel_version> build

8. Compile the module using the following commands:

   cd IEGD_Patches/IKM
   ./install.sh
   depmod -a
   modprobe iegd_mod

7.4.4 Using the IEGD Kernel Module

Note: This needs to be run after adding the IEGD components and updating the symbolic links in LINUX.

An installation script is provided to perform the installation. The script generates the Makefile together with the compilation environment for that particular kernel or distro. To install the IKM, run the shell script provided:

cd IEGD_Patches/IKM

./install.sh

The installation script detects the kernel version and points to the proper header files location before creating the Makefile. The script then calls the Make program to start the compilation process. After compilation is complete, the script tries to install the IKM. If the script is run from a normal user, it prompts for the superuser password before copying the generated file and then runs the following command to resolve module dependencies:

depmod -a

To insert the module into the kernel, run

modprobe iegd_mod

This will load all the modules that iegd_mod depends on before loading iegd_mod itself.

IKM installation requires a matching kernel source tree and a working Linux build system. Some of the programs require some additional libraries; for example, a hardlink is required by Fedora Core 5.

7.5 Uninstalling the IKM

To uninstall the IKM, run the install script described in Section 7.4.4 on page 115 with following argument:

./install.sh uninstall

This deletes the IKM file from kernel module location and invokes

depmod -a

to resolve dependencies for other module. A reboot might be required because IKM cannot be removed through the `rmmod` utility if a previous agpgart is part of the kernel image.

7.6 Configuration

IEGD auto-detects all device information necessary to initialize the integrated graphics device in most configurations. However, you can customize the IEGD configuration for any supported display by editing the X-server's configuration file (`XF86Config` or `xorg.conf`). Please refer to the `XF86Config(5x)` or `Xorg(5x)` man page for general configuration details. This section only covers configuration details specific to the IEGD.

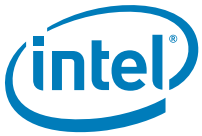
In order to configure the IEGD for Linux, you must edit the X server's configuration file (`XF86Config` or `xorg.conf`). You can either edit the configuration directly or can use CED to create configurations that must then be copied into the configuration file. Even if you use CED to create a configuration, you must still edit the Linux configuration file.

The following sections describe each method of configuring the IEGD for Linux.

7.6.1 Linux Configuration Using CED

You can configure the Linux driver settings using CED as described in Section 3.1, "IEGD Configuration Editor (CED)" on page 23 or in the CED online help.

The output file (yourbuildnamehere.x) from CED contains the settings required to configure the IEGD for Linux systems and can be pasted into the appropriate sections of the XF86Config or `xorg_conf` file.



7.6.2 Editing the Linux Configuration File Directly

Alternatively, you can edit the XF86Config or xorg.conf file directly. The following procedure outlines the steps to follow when editing the Linux configuration file. Section 7.6.3, "The Linux Configuration File" on page 117 provides details on each section of the configuration file.

1. Log in as root and open the configuration file for editing. The configuration file is typically located in the /etc/X11 directory but may be located elsewhere on your system.
2. In the Device section of the configuration file, enter the appropriate information for your driver. The configuration file must have at least one Device section. The Device section lets you define information about IEGD. You can use a single Device section for single, twin, or clone configurations. For Dual Independent Head configurations, you must specify a second Device section.
3. In the Screen section, enter information for each display in your configuration. The configuration file must have at least one Screen section. The Screen section binds a Device with a Monitor and lets you define resolution modes for the display. The Screen section is referenced in the ServerLayout section of the configuration file.
4. In the Monitor section, define monitor specifications and timings that will be used for the display. You must have a Monitor section defined for each display in your configuration. The Monitor section is referenced by the Screen section.
5. Save your changes to the file. For systems booted to run level 3, startx to restart. For systems booted to run level 5, kill X (Alt-backspace) to restart. Reboot if necessary.

7.6.3 The Linux Configuration File

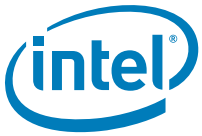
To configure the IEGD for use with Linux, you must edit the Linux configuration file (XF86Config or xorg.conf). There are several sections within the configuration that must be edited or created, including:

- Device Sections
- Screen Sections
- Monitor Sections
- ServerLayout Section (when configuring DIH)
- ServerFlags Section (when configuring Xinerama)

The above Sections are described following Figure 16. Please see the XF86Config or xorg.conf man pages for complete details. Figure 16 presents a sample XF86Config file.

Figure 16. Sample XF86Config File

```
# XFree86 4 configuration created by redhat-config-xfree86

Section "ServerLayout"
        Identifier     "Default Layout"
        Screen      0  "Screen0" 0 0
        InputDevice    "Mouse0" "CorePointer"
        InputDevice    "Keyboard0" "CoreKeyboard"
        InputDevice    "DevInputMice" "AlwaysCore"
EndSection

Section "Files"
# RgbPath is the location of the RGB database.  Note, this is the name of the
# file minus the extension (like ".txt" or ".db").  There is normally
# no need to change the default.
# Multiple FontPath entries are allowed (they are concatenated together)
# By default, Red Hat 6.0 and later now use a font server independent of
# the X server to render fonts.
        RgbPath      "/usr/X11R6/lib/X11/rgb"
        FontPath     "unix/:7100"
EndSection

Section "Module"
        Load  "dbe"
        Load  "extmod"
        Load  "fbdevhw"
        Load  "glx"
        Load  "record"
        Load  "freetype"
        Load  "type1"
EndSection

Section "InputDevice"
# Specify which keyboard LEDs can be user-controlled (eg, with xset(1))
#       Option  "Xleds          ""1 2 3"
# To disable the XKEYBOARD extension, uncomment XkbDisable.
#       Option  "XkbDisable"
# To customise the XKB settings to suit your keyboard, modify the
# lines below (which are the defaults).  For example, for a non-U.S.
# keyboard, you will probably want to use:
#       Option  "XkbModel""pc102"
# If you have a US Microsoft Natural keyboard, you can use:
#       Option  "XkbModel""microsoft"
#

# Then to change the language, change the Layout setting.
# For example, a german layout can be obtained with:
#       Option  "XkbLayout""de"
# or:
#       Option  "XkbLayout""de"
#       Option  "XkbVariant""nodeadkeys"
#
# If you'd like to switch the positions of your capslock and
# control keys, use:
#       Option "XkbOptions""ctrl:swapcaps"

# Or if you just want both to be control, use:
#       Option  "XkbOptions""ctrl:nocaps"
#
        Identifier  "Keyboard0"
        Driver      "keyboard"
        Option      "XkbRules" "xfree86"
        Option      "XkbModel" "pc105"
        Option      "XkbLayout" "us"
EndSection

Section "InputDevice"
        Identifier  "Mouse0"
        Driver      "mouse"
        Option      "Protocol" "IMPS/2"
        Option      "Device" "/dev/psaux"
        Option      "ZAxisMapping" "4 5"
        Option      "Emulate3Buttons" "no"
EndSection
```



```
Section "InputDevice"
# If the normal CorePointer mouse is not a USB mouse then
# this input device can be used in AlwaysCore mode to let you
# also use USB mice at the same time.

        Identifier  "DevInputMice"
        Driver      "mouse"
        Option     "Protocol" "IMPS/2"
        Option     "Device" "/dev/input/mice"
        Option     "ZAxisMapping" "4 5"
        Option     "Emulate3Buttons" "no"
EndSection

Section "Monitor"
        Identifier   "Monitor0"
        VendorName   "Monitor Vendor"
        ModelName    "Eizo T550"
        DisplaySize  290220
        HorizSync    30.0 - 82.0
        VertRefresh  50.0 - 160.0
        Option     "dpms"
EndSection

Section "Monitor"
        Identifier   "Monitor1"
        VendorName   "SamSung"
        ModelName    "SyncMaster 192MP"
EndSection

Section "Device"
        Identifier "Intel_IEGD-0"
        Driver     "iegd"
        VendorName "Intel(R) DEG"
        BoardName  "Embedded Graphics"
        BusID      "0:2:0"
        Screen      0
        Option     "PcfVersion"                    "1792"
        Option     "ConfigId"                      "1"
        Option     "ALL/1/name"                    "IEGD_Example"
        Option     "ALL/1/General/PortOrder"       "42000"
        Option     "ALL/1/General/DisplayConfig"   "1"
        Option     "ALL/1/General/DisplayDetect"   "0"
        Option     "ALL/1/General/CloneRefresh"    "60"
        Option     "ALL/1/General/CloneWidth"      "1280"
        Option     "ALL/1/General/CloneHeight"     "1024"
        Option     "ALL/1/Port/4/name"                     "Sample_LVDS"
        Option     "ALL/1/Port/4/General/EdidAvail"        "3"
        Option     "ALL/1/Port/4/General/EdidNotAvail"     "1"
        Option     "ALL/1/Port/4/General/Rotation"         "0"
        Option     "ALL/1/Port/4/General/Edid"             "1"
        Option     "ALL/1/Port/2/name"                     "Sample_SDVO"
        Option     "ALL/1/Port/2/General/EdidAvail"        "3"
        Option     "ALL/1/Port/2/General/EdidNotAvail"     "1"
        Option     "ALL/1/Port/2/General/Rotation"         "0"
        Option     "ALL/1/Port/2/General/Edid"             "1"
        Option     "PortDrivers"                           "lvds sdvo"
EndSection

Section "Screen"
        Identifier       "Screen0"
        Device           "IntelIEGD-1"
        Monitor          "Monitor0"
        DefaultDepth     16
        SubSection "Display"
                Depth   16
                Modes   "800x600" "640x480"
        EndSubSection
EndSection

Section "DRI"
        Group        0
        Mode         0666
EndSection
```

7.6.3.1 Device Section

The Device section provides a description of a graphics device. The Linux configuration file (XF86Config or xorg.conf) must have at least one Device section for the graphics driver. If your chipset supports multiple graphics pipelines, you may have multiple Device sections, but in most situations, only one is required. If you are creating a Dual Independent Head (DIH) configuration, you must have at least two Device sections.

The Device sections in the XF86Config and xorg.conf configuration files have the following format:

```
Section “Device”
       Identifier “devname”
       Driver “iegd”
       ...
EndSection
```

The Identifier field defines the device. This name is used to associate the device with a screen and is referenced in Screen sections.

The Driver field defines the driver to use and is a required field in the Device section. The intel driver, intel_drv.o, must be installed in the /usr/X11R6/lib/modules/drivers directory.

The remainder of the Device section can contain IEGD-specific options. Please see Table 23 on page 122 for a list and description of IEGD supported options.

7.6.3.2 Screen Section

The Screen section is used to bind a Screen with a Device and a Monitor. It is also used to define resolution modes, color depths, and various other screen characteristics. Please see the XF86Config or xorg man page for detailed information.

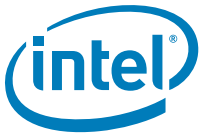
The Screen section has the following format:

```
Section “Screen”
       Identifier    “screenname”
       Device        “devname”
       Monitor       “Monitor0”
       DefaultDepth24
              Subsection “Display”
                     Depth  24
                     Modes  “1280x1024” “1024x768” “800x600” “640x480”
              EndSubSection
EndSection
```

7.6.3.3 Monitor Section

Use the Monitor section to define monitor characteristics and timings for a display. You should have one Monitor section for each display your system supports. The Monitor section is referenced in a Screen section. And has the following format.

```
Section “Monitor”
       Identifier “Monitor0”
       VendorName “NEC”
       MonitorName “MEC MultiSync LCD”
       HorizSync    30-60
       VertRefresh  50-75
       ...
EndSection
```



7.6.3.4 ServerLayout Section

The `ServerLayout` section defines the overall layout of the system configuration. Input devices are specified in the InputDevice fields and output devices usually consist of multiple components (such as a graphics board and a monitor, which are bound together in a Screen section). You typically only need to edit this section when you are using a DIH configuration. You need to add a line to reference the second `Screen` section and specify its relative location to the first screen. In the following sample, the line beginning with `Screen 1...` is required for DIH configurations.

```
Section “ServerLayout”
       Identifier “Default Layout”
       Screen 0 “Screen0” 0 0
       Screen 1 “Screen1” RightOf “Screen0”
       InputDevice entries...
EndSection
```

7.6.3.5 ServerFlags Section

If you are configuring the IEGD for Xinerama support, you must set the “`Xinerama`” option to “`True`” in the `ServerFlags` section of the configuration file.

```
Section “ServerFlags”
       Option “Xinerama” “True”
EndSection
```

7.6.4 XFree86 and Xorg Configuration Options

The IEGD provides a format syntax for Linux configuration options. The syntax is similar to the Microsoft Windows* INF file and is as follows:

```
       “All/<ConfigID>/<block name>/<option name>”
```

The IEGD parses the configuration options and looks for “new-style” 4.0 and later options. If it doesn't find any, then it falls back to processing old-style options.

The device configuration must contain the “pcfversion” option with value “0x700”. This indicates to the driver the options format to use. Earlier pcfversions (0 and 0x400) are supported for backward compatibility.

The IEGD driver supports multiple sets of installed configuration options that may be selected at runtime.

Configuration ID 0 is used unless otherwise specified in the configuration file or supplied by the system BIOS.

Table 23 shows the supported driver options:

Table 23. Supported Driver Options (Sheet 1 of 2)

Option	Description
Option “PcfVersion” “integer”	This option indicates that the new IEGD format is being used for the Linux Configuration files (`XF86Config` or `xorg.conf`). The new format is hierarchical (similar to the Microsoft Windows* INF file) and allows both global and per-configuration information to be stored in the X-server’s configuration file (`XF86Config` or `xorg.conf`) rather than having per-configuration information stored separately in the `EDIDx.bin` file. This option is usually set to 0400 hex (1024 decimal) and is required for the new format.
Option "SWCursor" "boolean"	Enable the use of the software cursor. The default is off and the hardware cursor is used.
Option "ShadowFB" "boolean"	Enable or disable double buffering on the framebuffer. The default is that double buffering is disabled.
Option "TearFB" "boolean"	Disable or enable wait for vblank when doing blits. The default is to not wait for vblank when doing blits. This is faster but may cause visible tearing of the display. Set to “1” (default) to not wait for vblank Set to “0” to wait for vblank to reduce tearing
Option "XVideo" "boolean"	Disable or enable XVideo support. In a dual independent head configuration, either the first display or the second display support XVideo. Both displays cannot support XVideo simultaneously. The default is XVideo support is enabled.
Option "XVideoBlend" "boolean"	Disable or enable XVideo support using the 3D blend manager. This provides XVideo support in configurations that cannot be supported with overlay. For example, this is supported on both displays in a dual independent head setup. It is also supported when the display is rotated or flipped. Color key is only supported if ShadowFB is enabled and the VideoKey is defined. The default is XVideoBlend support is enabled.
Option “ConfigID” “integer”	This option identifies the configuration.
Option “All/<ConfigID>/Name” “string”	A quoted string used to identify the configuration name.
Option "All/<ConfigID>/Comment" "string"	A quoted string used to identify the configuration file. Comment is a required field for Linux configurations.
Option "PortDrivers" "string"	This option specifies which port driver(s) must be loaded. The list is a space- or comma-separated list of port driver names corresponding to the *.so port driver files included with the Linux version of the driver. You may specify multiple port driver names if there are multiple DVO devices that are currently on the system or may potentially be used. For example, specify "th164 sii164" to load port drivers for Thine 164 and Silicon Image 164* DVO devices. The port driver for the built-in analog output from the GMCH is always included and does not need to be specified in the PortDrivers option. The port drivers for the built-in LVDS and TV components (on chipsets with such features) are NOT automatically included. The "lvds" and "inttv" port drivers must be specified in order to use those output ports.

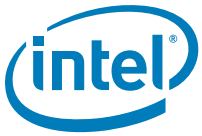


Table 23. Supported Driver Options (Sheet 2 of 2)

Option	Description
Option "All/<ConfigID>/General/PortOrder" "string"	This option can be used to change the default port allocation order. The default order can vary depending on chipset. List the port type numbers in the priority order starting from first to last. The port type numbers are as follows: 1 - Integrated TV Encoder (mobile chipsets only) 2 - DVO/sDVO B port 3 - DVO/sDVO C port 4 - Integrated LVDS port (mobile chipsets only) 5 - Analog CRT port To set the order as Integrated TV Encoder, ANALOG, LVDS, DVO C, DVO B set the PortOrder string to "15432". Zeros can be used to specify don't care. Setting this option incorrectly can result in port allocation failures.
Option "All/<ConfigID>/Port/<port number>/General/Rotation" "integer"	Rotate the display. Valid values are 0, 90, 180, 270.
Option "All/<ConfigID>/Port/<port number>/General/Flip" "boolean"	Invert the display horizontally.
Option "All/<ConfigID>/Port/<port number>/VideoKey" "integer"	This sets the color key for XVideo and XVideoBlend. This value is either a 24-bit value or a 16-bit value, depending on the pixel depth of the screen. The color key is always enabled for XVideo, even when it is not defined. The color key is always disabled for XVideoBlend unless both this option is defined and the ShadowFB option is enabled. The default color key for XVideo is 0x0000ff00. For XVideoBlend, the color key is disabled by default.
Option "All/<ConfigID>/Port/<port number>/CloneWidth" "integer"	This sets the display width for a clone port when CloneDisplay is active. The default is 640.
Option "All/<ConfigID>/Port/<port number>/CloneHeight" "integer"	This sets the display height for a clone port when CloneDisplay is active. The default is 480.
Option "All/<ConfigID>/Port/<port number>/CloneRefresh" "integer"	This sets the display vertical refresh rate for a clone port when CloneDisplay is active. The default is 60 Hz.
Option "All/<ConfigID>/Port/<port number>/AttributePath" "string"	Specify the directory path where attribute data will be saved and restored from. If this is not set, then attribute data will not be saved/restored. The default is no directory path.
Option "All/<ConfigID>/Port/<port number>/Active" "boolean"	Make the device ports active and enabled on startup. This is the default. When this option is set to FALSE, the driver will allocate the output ports for this device but will not set the mode or enable the output. To enable the output, the port control extension must be used.
Option "All/<ConfigID>/Port/<port number>/EDID" "boolean"	Enable or disable reading of EDID data from the output port device. Note that if the EDID option is specified in the config file (`XF86Config` or `xorg.conf`), all per-port EDID options in the configuration are overwritten by the EDID option specified in the config file.

7.6.5 Sample Dual Independent Head (DIH) Configuration

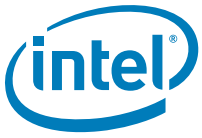
For dual independent head operation, several additional options must be set in the Device sections for each head. Both Device sections must specify the BusID, and the BusID must be the same for both devices. Each Device section must specify the Screen section that will associate the device with the Screen option.

```
BusID  - B:F:S (Bus, Function, Slot)
Screen  - number
```

Figure 17 shows a sample DIH configuration. Only the Device, Screen, and Server Layout sections of the configuration file are shown. For a complete example of a configuration file, see Figure 16 on page 118.

Figure 17. Sample DIH Configuration

```
Section "Device"
    Identifier "IntelEGD-1"
    Driver     "iegd"
    BusID       "0:2:0"
    Screen      0
    VideoRam    32768
EndSection

Section "Device"
    Identifier  "IntelEGD-2"
    Driver      "iegd"
    BusID       "0:2:0"
    Screen      1
    VideoRam    32768
EndSection

Section "Screen"
    Identifier  "Screen 1"
    Device      "IntelEGD-1"
    Monitor     "Monitor1"
    DefaultDepth 24

    Subsection "Display"
        Depth       8
        Modes       "1280x1024" "1024x768" "800x600" "640x480"
        ViewPort    0 0
    EndSubsection

    Subsection "Display"
        Depth       16
        Modes       "1280x1024" "1024x768" "800x600" "640x480"
        ViewPort    0 0
    EndSubsection
    Subsection "Display"
        Depth       24
        Modes       "1280x1024" "1024x768" "800x600" "640x480"
        ViewPort    0 0
    EndSubsection EndSection

Section "Screen"
    Identifier  "Screen 2"
    Device      "IntelEGD-2"
    Monitor     "Monitor2"
    DefaultDepth 24
    Subsection "Display"
        Depth       8
        Modes       "1280x1024" "1024x768" "800x600" "640x480
        ViewPort    0 0
    EndSubsection
    Subsection "Display"
        Depth       16
        Modes       "1280x1024" "1024x768" "800x600" "640x480"
        ViewPort    0 0
    EndSubsection
```



```
            Subsection "Display"
                Depth       24
                Modes       "1280x1024" "1024x768" "800x600" "640x480"
                ViewPort    0 0
            EndSubsection
EndSection

        Section "ServerLayout"
            Identifier  "Dual Head Layout"
            Screen "Screen 1"
            Screen "Screen 2" Right Of "Screen 1"
            InputDevice "Mouse1" "CorePointer"
            InputDevice "Keyboard1" "CoreKeyboard"
        EndSection
```

7.6.6 Video Memory Management

The Intel integrated graphics controllers have a unified memory architecture that uses system memory for video RAM. The amount of memory used is not fixed and depends on the configuration. Address space for framebuffers and back buffers is always reserved, along with some scratch space for 2D and 3D acceleration. Use of some features such as Video Overlay also affect video memory allocations. The amount of video memory allocated for pixmap surfaces is configurable. By default, 16 Mbytes is used for each screen. You can change this with the `VideoRam` option in the `Device` section of the configuration file. It may be set to any reasonable value up to 32 Mbytes. Increasing this value reduces the amount of system memory available for other applications. This value is in units of 1024 Kbytes (32 Mbytes is represented by 32768).

7.6.7 Graphics Port Initialization

When used with a graphic chipset that supports multiple graphics pipelines, the driver supports multiple screens and Xinerama. This support is enabled by creating additional `Device` sections for each additional graphics device on the PCI bus. The driver locates the first device on the bus and associates it with the device section that matches (or one that doesn't specify a busID). This becomes the primary display. If the chipset supports multiple display pipes, and the config file specifies two `Device` sections and two `Screen` sections, the driver attempts to operate in a dual-independent head mode. Once all the graphics devices and device sections have been matched up, the driver attempts to allocate any remaining output ports and attach them to the primary graphics device. For example:

- Two pipes and two ports allows for dual independent displays.
- One pipe and two ports allows for a cloned display (915GV special case).

7.6.8 OpenGL Support

The IEGD supports OpenGL* for the following Intel chipsets:

- Mobile Intel® GME965 Express chipset
- Intel® Q965 Express chipset
- Mobile Intel® 945GME Express chipset
- Intel® 945G Express chipset
- Mobile Intel® 915GME Express chipsets
- Intel® 915GV Express chipsets
- Mobile Intel® 910GMLE Express chipset
- Intel® 855GME chipset
- Intel® 852GME chipset
- Intel® 852GM chipset

The OpenGL implementation for IEGD consists of three components.

- `libGL`: This is the shared library that implements the OpenGL and GLX APIs. It is linked by all OpenGL applications.
- `iegd.ko`: This is the Direct Rendering Manager (DRM). It is a kernel module that provides the OpenGL application with the permissions necessary to directly access the DMA buffers used by libGL.
- `X Server`: The existing IEGD X server driver has been enhanced to communicate with `libGL`.

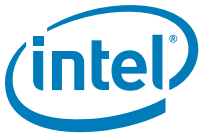
Installing the IEGD OpenGL driver provides a fully hardware accelerated implementation of the OpenGL library to applications. This implementation makes use of a Direct Rendering technology, which allows the client to directly write to DMA buffers that are used by the graphics hardware.

Due to the use of direct rendering technology, system designers should take special care to ensure that only trusted clients are allowed to use the OpenGL library. A malicious application could otherwise use direct rendering to destabilized the graphics hardware or, in theory, elevate their permissions on the system.

A system designer can control the access to the direct rendering functionality by limiting the access to the DRI device file located at:

```
/dev/dri/card0
```

The permissions on this device are set by the X-Server using the information provided in the “DRI” section of the configuration file.



7.6.8.1 OpenGL Installation

To install the IEGD libGL onto a system, copy the `libGL.so` binary from the package to the standard location. Then compile and install the Direct Rendering Manager (DRM) kernel module from the sources provided. Lastly, enable the DRI option in the X server's configuration file.

Note: The system likely has a version of libGL already installed. You may want to make a backup copy of the existing library before installing the IEGD library. There are two libGL binaries included in the package that each support different hardware types. Only the one supporting target hardware should be installed. Chipsets of generation 3 and earlier (945 and earlier) should use libGLgn3.so and generation 4 chipsets (965 and above) should use libGLgn4.so.

1. cd IEGD_x_Linux/Driver/Xorg-X11R6.8/
2. cp libGLgn#.so /usr/lib/libGL.so.1.2
   - 85x, 910GMLE, 915GME, 915GV, 945GME, 945G
     Use libGLgn3.so
   - Q965, GME965
     Use libGLgn4.so
   - Q35 not supported in 8.0 (no 3D)
3. cd /usr/lib
4. ln -sfv /usr/lib/libGL.so.1.2 libGL.so
5. ln -sfv /usr/lib/libGL.so.1.2 libGL.so.1
6. ldconfig

Note: **Skip steps 7, and 8 if you are using the IKM method detailed in Section 7.4.**

7. cd IEGD_Patches/Drm
8. make
   This will build and install the kernel module for the currently running kernel. If another kernel is installed or used, this step must be performed again.
9. depmod -Ae
10. Restart the X-Server or restart the system.

7.6.8.2 OpenGL Use Considerations

Allocation of Mipmaps and Memory Usage

Under normal circumstances the OpenGL driver will allocate all mip levels for a texture at allocation time. This is due to the fact that the OpenGL API allows an application to make use of the mips without first conveying an intention to do so. All mips are therefore available all the time.

The IEGD OpenGL driver has a special-case scenario to prevent the allocation of mips when the application can ensure that they will never be populated or used. On some hardware configurations this can save 50% on texture memory usage. To enable this feature, the application should do the following:

Using glTexParameter*(), set the GL_TEXTURE_MAX_LEVEL parameter to 0 before populating the texture (before any call to glTexImage2D()). This will prevent mips 1-N from being allocated but will not prevent them from being used. If the mips are inadvertently used, the results are undefined.

7.6.9 Sample Advanced EDID Configurations for Linux OS

The edid_avail and edid_not_avail parameters control the available timings for any display. The edid_avail parameter is used when EDID information is read from the display. If the driver is unable to read EDID information from the display or if the edid parameter in the .pcf file is set to "0" (disable), then the settings of the edid_not_avail parameter are used. Please see Section 3.1, "IEGD Configuration Editor (CED)" on page 23 and CED online help.

There is an edid option that can be placed in the XF86Config or xorg.conf files that controls the behavior of the overall driver, and there are also EDID settings within CED that control the behavior on each port (edid, edid_avail, and edid_not_avail). The combination of these settings determine how the driver behaves. Table 24 shows various configurations and the expected behavior of the driver.

Table 24. Sample Advanced EDID Configurations for Linux OS

Case	XF86Config "edid" option	CED: Per port "edid" option	Expected driver behavior
1.	No "edid" option specified	No edid flag specified	For every port, driver uses edid_avail.
2.	No "edid" option specified	edid=0 for some ports and edid=1 for some ports	For edid=0 ports, driver uses edid_not_avail flags. For edid=1 ports driver uses edid_avail flags.
3.	"edid"=no	Setting does not matter.	For all ports driver will not read edid and interprets edid_not_avail flags. Driver overrides any per-port pcf edid flags, treats all displays as EDID-less displays, and uses edid_not_avail flags.
4.	"edid"=yes	edid=0 for some ports and edid=1 for some ports	Same as case 2

Note: For all cases:

1. If there is not an edid_not_avail flag specified for a port, and an EDID-less display is detected, the driver defaults to using the standard built-in timings for that port.

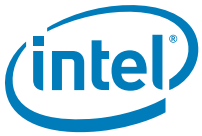
2. If there is not an edid_avail flag specified for a port, and an EDID display is detected, the driver defaults to using the EDID data from the display, plus any user specified DTDs.

3. If edid=1 and the display device is EDID-less, the driver uses edid_not_avail flags.

7.6.10 AGP GART Errors

The following are the most common AGP GART errors:

1. Symptom: No "agpgart: " in the system log
   Cause: The IEGD AGP GART patch has not been applied to the system.
2. Symptom: The Xorg.0.log has the following:
   (EE) INTEL(0): gart.c: Acquire IOCTL failed
   Cause: The IEGD AGP GART module has not been loaded.



7.7 Runtime Operation

7.7.1 Runtime Configuration GUI (iegdgui)

You can change the configuration and runtime attributes of the driver using the iegdgui runtime configuration tool. The `iegdgui` resides in the `/Utilities` directory. The Intel Embedded Graphics Drivers GUI (iegdgui) is a GUI application that is used to view and control the Intel Embedded Graphics Drivers. This tool is used to retrieve status information of the display and driver and also to configure the supported display attributes. The iegdgui also demonstrates multi-monitor support.

7.7.1.1 iegdgui Setup

To run the `iegdgui`, you need to ensure that the XServer has been configured to use the IEGD. See Section 7.6, “Configuration” on page 116 for details on configuring and installing the IEGD.

You need GTK+ and libglade, which are part of the Linux distribution and should already be installed.

Ensure that the LD_LIBRARY_PATH environment variable points to the X11R6 library. If it does not, type the following command:

export LD_LIBRARY_PATH=/usr/X11R6/lib

1. Ensure the `iegdgui` is executable by changing directories to .../`IEGD_X_x_Linux/Utilities` (where X is the major version and x is the minor version) and running the following command:

   ls -l iegdgui

   Executable permissions should be set for all three Linux groups (user, group, world) and should look like this:

   rwxr_xr_x iegdgui ...

   If the permissions do not contain an “x” for each group, change the permissions using the following command:

   chmod +rwx iegdgui

After you have completed this step, the IEGD can be launched.

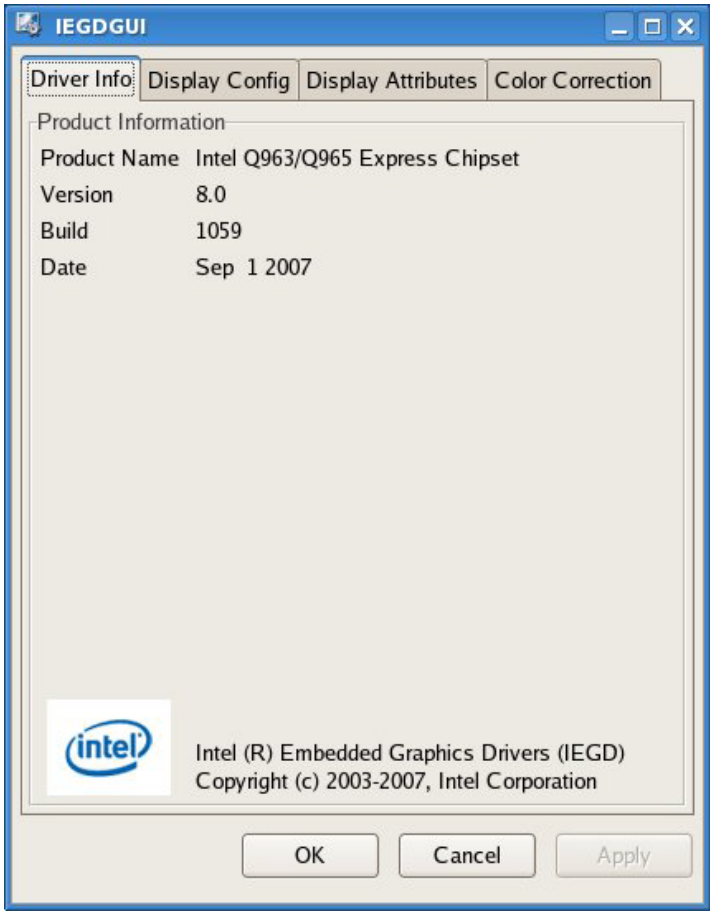
7.7.1.2 Using the iegdgui Runtime Configuration Utility

The iegdgui application provides four tabs: **Driver Info**, **Display Config**, **Display Attributes,** and **Color Correction**.

- **Driver Info**: Contains the driver information.
- **Display Config**: Contains current display information and allows configuration of display configurations, display resolutions for primary and secondary displays and enabling/disabling of a specified port.
- **Display Attributes**: Contains the supported Port Driver (PD) attributes and allows configuration of PD attributes.
- **Color Correction** — Contains current color-correction information for the framebuffer and overlay. Using this tab, you can change the framebuffer and overlay color settings.

Figure 18 shows the **Driver info** tab.

Figure 18. Example Linux Runtime Configuration GUI — Driver Info Tab

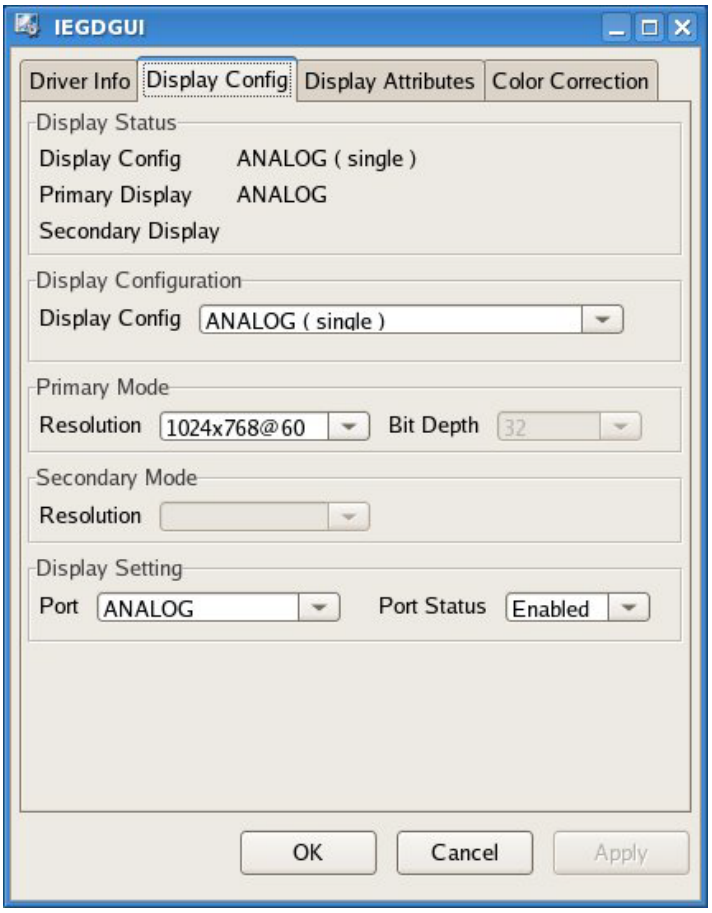


To view current display information and or to change the current configuration of display configurations, display resolutions of the primary and secondary displays and enabling/disabling of a specified port, click the **Display Config** tab.

Note: If you make any changes to the configuration, click **Apply** for the changes to take effect.

Figure 19 shows a sample configuration.

Figure 19. Example Linux Runtime Configuration GUI — Display Config Tab



The **Display Status** section of the above dialog shows the current configuration for the **Primary** and **Secondary** displays.

In the **Display Configuration** section of the dialog, select the required display configuration in the **Display Config** drop-down list. This allows the user to choose between Single, Twin, Clone and Extended for all connected ports. A maximum of two ports per display configuration is currently allowed.

In the **Primary Mode** and **Secondary Mode** sections of the dialog, you can change display resolutions via the **Resolution** drop-down list.

In the **Display Settings** section of the dialog, you can view and change the settings for a port and then rotate and flip the display via the appropriate drop-down lists:

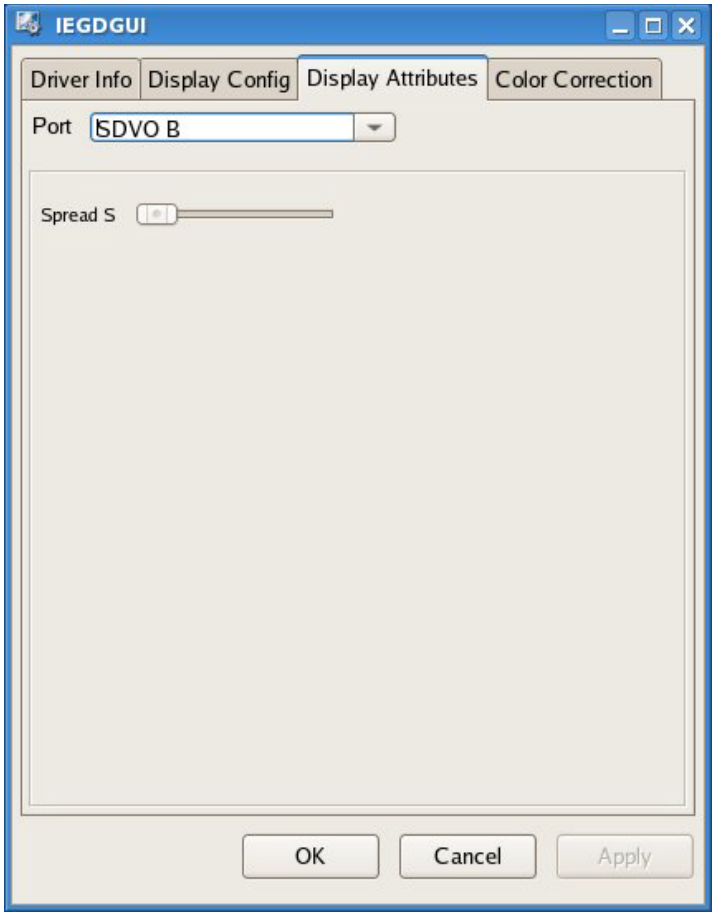
- **Port**: Allows you to select the required port.
- **Port Status**: Allows you to enable or disable the selected port.

Note: If you change any configuration settings in the **Display Config** dialog box, click **Apply** for the changes to take effect.

To view or change the attributes for a port, click the **Display Attributes** tab. Figure 20 shows a sample configuration. Please see Appendix B for detailed information on port driver attributes.

Note: If you make any changes to the port driver attributes, click **Apply** for the changes to take effect.

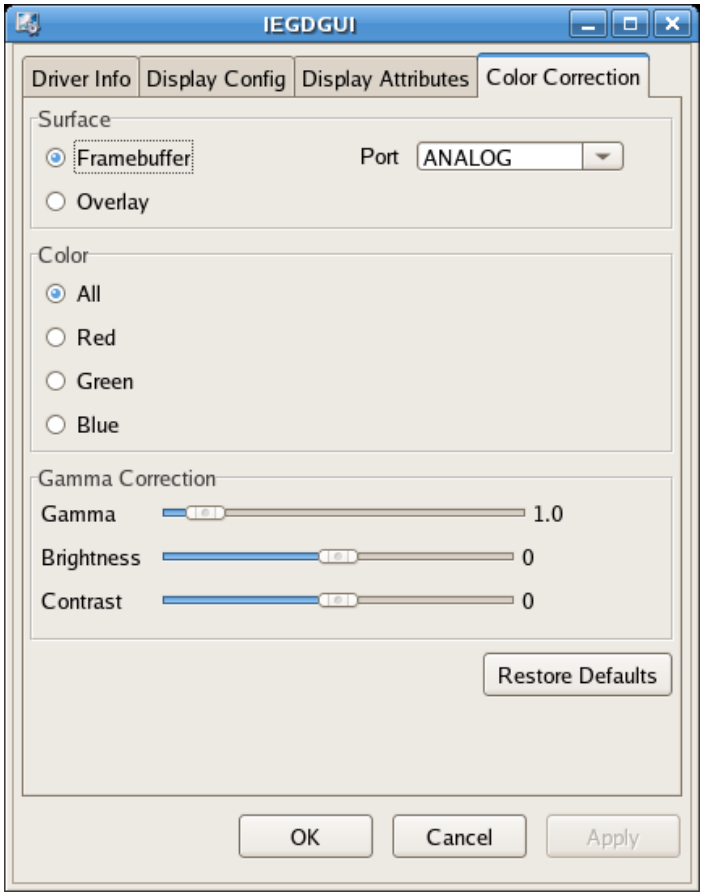
Figure 20. Example Linux Runtime Configuration GUI — Display Attributes Tab



To view and change color corrections, click the **Color Correction** tab. Figure 21 and Figure 22 show sample Color Correction tab screens for **Framebuffer** and **Overlay**, color correction values for which are shown in Table 18 and Table 19.

Note: If you make any changes to the color-correction attributes, click **Apply** for the changes to take effect.

Figure 21. Example Linux Runtime Configuration GUI — Color Correction Tab (Framebuffer)



The following steps present an example color-correction procedure:

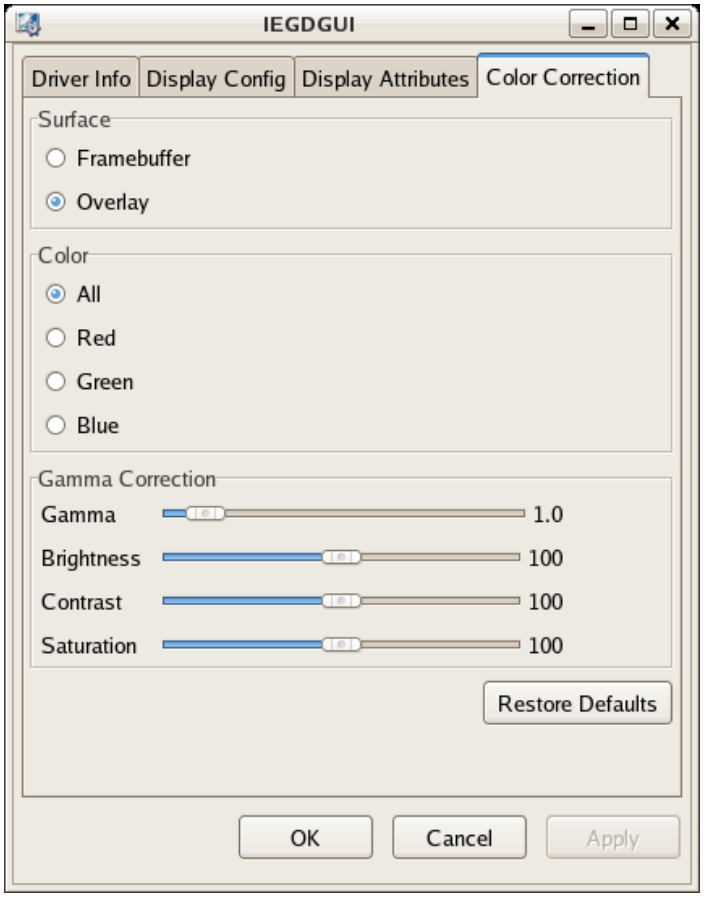
a. Select **Framebuffer** in the **Surface** section and select the appropriate port for the color correction to be applied to or select **Overlay** in the Surface section for color correction to be applied to the overlay.

b. Select the required color to be corrected in the **Color** section.

c. Select the required color attribute to be corrected in the **Gamma Correction** section.

d. Click **Restore Defaults** to restore the default values.

Figure 22. Example Linux Runtime Configuration GUI — Color Correction Tab (Overlay)



7.8 Enabling Damn Small Linux

This section describes IEGD enablement of Damn Small Linux* (DSL). For further DSL-specific information and downloads, please see http://www.damnsmalllinux.org.

This section discusses the following IEGD/DSL-related topics:

- Damn Small Linux introduction
- XFree86 versus TinyX
- Running IEGD on Damn Small Linux
- RAM size
- Shrinking and modifying the extension

7.8.1 Damn Small Linux Introduction

DSL is intended for “liveCD” or “frugal” installs, i.e., installed on a USB flash drive for booting, which should only require 50 Mbytes. You can also experiment with a hard-disk installation. Note that a minimal, 50-Mbyte flash drive installation is compressed and loaded at boot time. This “original” DSL filetree is permanent and always gets restored after each reboot. In addition, the compressed filetree is always located in the first partition of the flash drive.

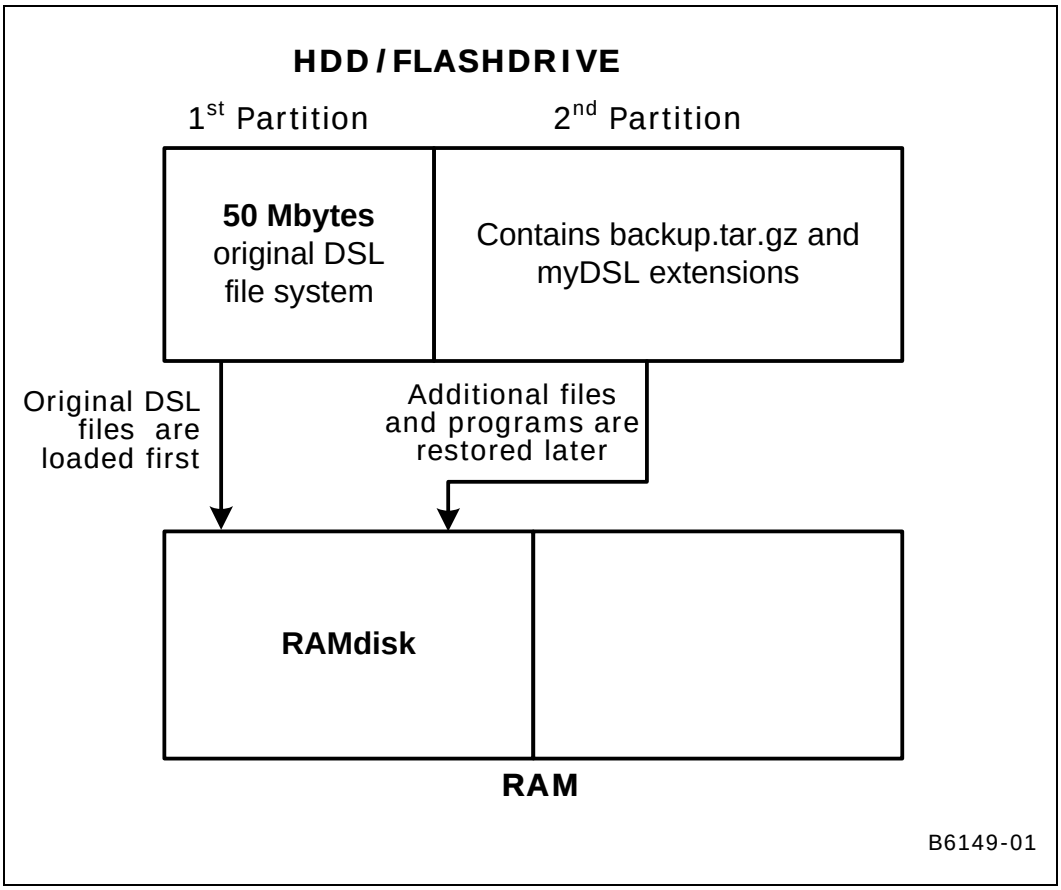
7.8.1.1 Extending Damn Small Linux

A second USB-drive partition (sda2) can be used for additional programs or files for use with DSL; see Figure 23. An external partition or CD-ROM can also be used for such additional items. Note that for users to write or create new files, DSL uses a RAMdisk, which means that at boot time a certain portion of the RAM is reserved as disk space.

There are two main methods of adding items to DSL:

- **myDSL** — an extension package that adds new programs to DSL at boot time. Please see the myDSL page (http://distro.ibiblio.org/pub/linux/distributions/damnsmall/mydsl/) for extensions and installation instructions.
- **Backup restore script** in the original DSL file system.

Figure 23. Damn Small Linux Partition



7.8.2 XFree86 Versus TinyX

DSL uses TinyX* by default. To use XFree86* instead, see “XFree86.dsl.info” in the XFree86.dsl package (http://distro.ibiblio.org/pub/linux/distributions/damnsmall/mydsl/system/XFree86.dsl). Two changes, which are outlined in XFree86.dsl.info, are required to enable XFree86.

Note: IEGD works only with the optional XFree86 X server, not the default TinyX X server.

7.8.3 Running IEGD on Damn Small Linux

Once the XFree86 package has been installed, you can install and use the IEGD driver. The first thing to do is to try to get the agpgart compiled before running XFree. The current DSL version, 2.3, uses kernel 2.4.26. The agpgart patch 2.4.24 (agpgart.patch-2.4.24) is also used. Kernel source for 2.4.26 can be found on http://www.kernel.org/.

Due to the limited space/resources that a DSL system may have, it is advisable to compile the agpgart in another distro before transferring it into DSL. Remember also to disable version information in make menuconfig when you attempt to compile in an external distro. During enablement of DSL, a RH9 distro is used to patch and compile the agpgart in kernel 2.4.26.

Using RH9 as an example, execute the following commands:

1. Copy linux-2.4.26.tar.gz to /usr/src/
2. cd /usr/src
3. tar xzvf linux-2.4.26.tar.gz
4. cp /usr/src/linux2.4.20-8/configs/kernel-2.4.20-i686.config ./.config
5. cd /usr/src/linux-2.4.26
6. Apply patch agpgart.patch.2.4.24
7. Make clean
8. make mrproper
9. cp ../.config ./.
10. Make menuconfig.

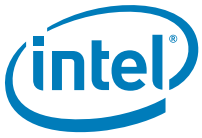
Note: For this step, remember to turn off version information under the kernel loadable module section. Also check that agpgart is being compiled as a modules here.

11. make dep make bzImage modules
12. Copy agpgart from /usr/src/linux-2.4.26/drivers/char/agp/agpgart.o
13. Transfer agpgart to DSL /lib/modules/2.4.26/kernel/drivers/char/agp/agpgart.o

After the agpgart has been modprobed, copy all the IEGD files to the proper directory. Startx and IEGD should be running then.

Check that IEGD is running by looking through the XFree logfile. It should be under /var/log/. Also, “Intel Embedded Graphics Driver” should display.

After all this done, you can ensure that IEGD is running after it has been rebooted. Add the appropriate files for IEGD into .filetool.lst include the agpgart.o in /lib/modules/2.4.26/kernel/drivers/char/agp run filetool.sh backup to backup the files. Next, change /opt/bootlocal.sh to modprobe the agpgart at startup.



7.8.4 RAM Size Constraint

A 128M RAM can be used if a swap file is created in the second partition. To make a swap file:

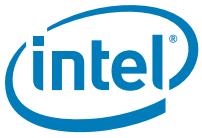
1. Make a blank file of fixed size using dd. Example: dd if=/dev/zero of=knoppix.swp bs=1M count=70
2. Make the file into a swap file. mkswap knoppix.swp
3. Turn on the swap. swapon knoppix.swp

Note: Remember to turn on the swap file every time you start DSL by changing the bootup script in /opt/bootlocal.sh.

7.8.5 Shrinking and Modifying the Extension

There are several types of extensions being used for myDSL. For more information on the different types, how to use them, and their creation, please refer to the myDSL wiki. Basically, myDSL extensions are tarballs that get untarred during bootup. For the purpose of enabling IEGD, the XFree86.dsl extension can be modified and shrunk to help save hard disk space.

This page is intentionally left blank.



8.0 Legacy VBIOS

8.1 Overview

In general, a legacy VGA VBIOS provides firmware-based graphics device initialization and rudimentary display output support independent of operating systems, such as text mode support. Operating system independence is accomplished through the legacy IBM* PC method of calling software interrupts, which the system BIOS sets up. Operating systems, prior to display drivers loading, use the rudimentary display output services provided by VBIOS to display data such as informational messages and splash screen. If no display drivers are discovered, operating system and/or applications may rely on the VBIOS during regular operation. In addition, during pre-operating system boot, the system BIOS may also rely on the rudimentary display output services from VBIOS to display items such as informational messages, splash screen, and user setup screen. The Intel Embedded VBIOS provides these rudimentary services by default.

For extended resolutions and industry-standard services, an independent standards organization, the Video Electronics Standards Association (VESA) has defined a set of VBIOS Extensions (VBEs) for the legacy VGA video BIOS. Like the legacy VGA video BIOS interfaces, these extensions are also accessible through the IBM PC method of calling software interrupts. In addition, the latest core standard (VESA/VBE Core 3.0) defines a method for accessing video BIOS functions through 16-bit Protected mode. The system BIOS, operating systems and/or applications can use these extended resolutions and services. The legacy Intel Embedded VBIOS supports the VESA/VBE Core 3.0 interfaces.

Some information may exist on a system utilizing the Intel Graphics Controller that is not accessible through any of the standard defined interfaces, such as the legacy VGA Video BIOS interface or the VESA/VBE interfaces. An example of information that cannot be retrieved through any of the standard defined interfaces is what displays are currently attached and detected. Also, some functions may not be accessible through any of the standard interfaces. An example of a function that cannot be accomplished through any standard interface is switching between the displays attached to the Intel graphics controller. For services such as these that are unique to the Intel Graphics Controller hardware, a set of Intel-defined functions exist. These functions are accessible through the legacy IBM PC method of calling software interrupts. Because these Intel functions are not standardized, they can only be used by custom applications or system BIOS.

During system initialization, the video BIOS may require from the system some information that it cannot retrieve on its own. On typical systems, this necessary information is available through the system BIOS. The video BIOS may retrieve the information from system BIOS through Intel-defined System BIOS Hook Functions. For example, if a system uses a TV as its display, it may use a jumper to indicate NTSC or PAL. The system BIOS can determine the setting of the jumper, then video BIOS will retrieve the setting from system BIOS using the appropriate System BIOS Hook Function. The only purpose for these system BIOS Hooks is to pass information between the Intel Video BIOS and the system BIOS; the functions are not accessible to any external applications. The Intel Embedded VBIOS supports these System BIOS Hook Functions. Section 8.3.1 describes how to turn on and turn off the system BIOS Hook Functions through configuration options.

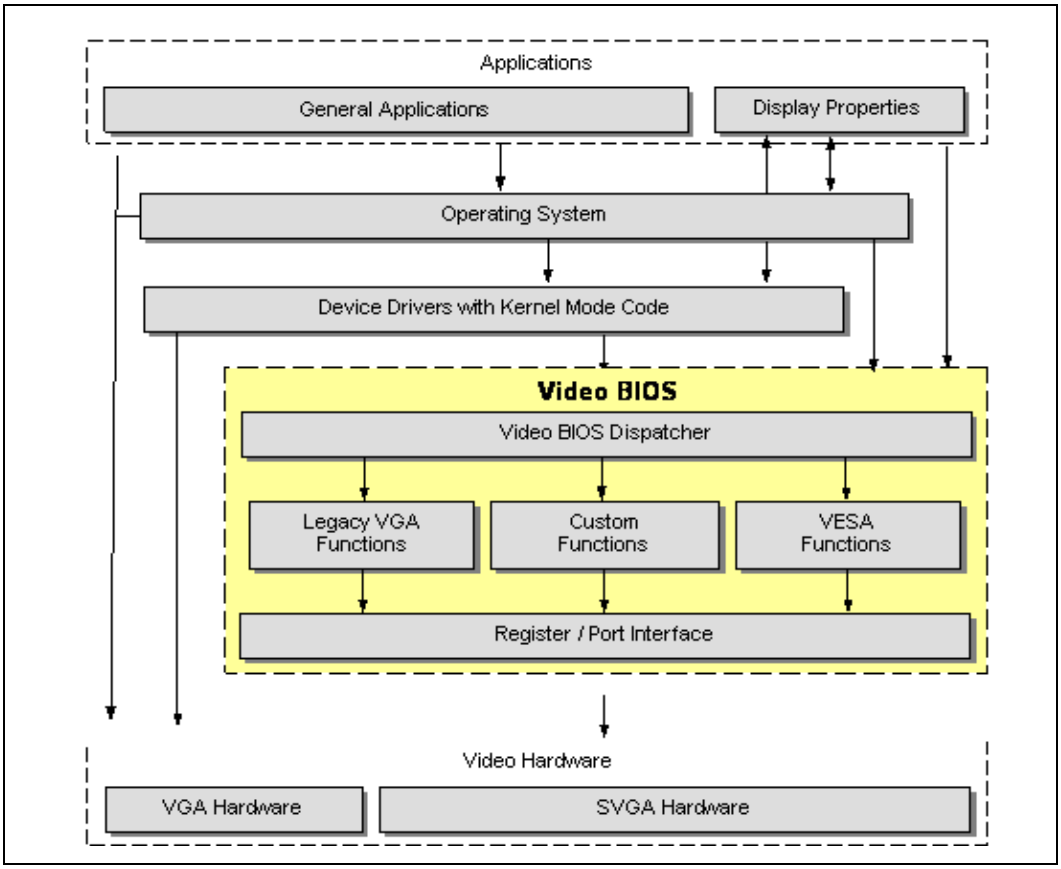
Some of the system details the VBIOS discovers can be useful for graphics drivers. For example, VBIOS may have already detected attached displays; that information could be passed to the driver so that rediscovery is not necessary. The Intel Embedded VBIOS uses graphics controller registers to pass data to the Intel® Embedded Graphics Drivers. This is intended to be used as VBIOS-to-driver communication only.

Due to the broad range of potential system configurations, it is not possible to create and provide a single VBIOS binary that can accommodate and automatically detect all different configurations while still fitting within the 64 Kbytes maximum VBIOS size required by customers. Binaries built with basic configurations are already included with the release package, but these work only with specific system configurations; see the Release Notes in the release package for details. For systems that do not fall within the default system definition, the VBIOS requires a configuration effort to accurately reflect the system configuration. The release package provides a set of tools and configurations that enable creating a new VBIOS binary tailored to the system. Features such as DVO devices, panel type, panel timings, new mode timings, I^2C ports, and more may be configured using the tools provided in the release package. Section 8.3.1 describes how to use the tools and the different configuration options.

8.1.1 VBIOS Model

The VBIOS provides functionality that abstracts the hardware for applications and operating systems. The VBIOS does not actively monitor access to the Intel Graphics Controller hardware registers, nor can it prevent other software from accessing the hardware directly. Often, the VBIOS assumes that it has sole control over the hardware registers; for instance, when a particular mode is set through the VBIOS and another application resets all timing registers for a different mode. Then, the VBIOS will assume that the original mode is still in effect. Therefore, it is highly recommended that applications and System BIOS use the VBIOS functions whenever possible.

Figure 24. VBIOS Model



8.2 Panel Detection

VBIOS accomplishes panel detection through an interface call to system BIOS. This is used to isolate the correct DTD in the table compiled into VBIOS. During boot, VBIOS uses this timing for setting initial graphics modes. Upon transition from system BIOS to OS, the panel ID is transferred to the graphics driver through a semaphore. As with VBIOS, the driver uses the ID to traverse its DTD table located in the registry. The correct DTD is then used to program the panel with the correct display mode.

The VBIOS relies on system BIOS to report the correct panel type associated with the system. This is conducted via an INT15 system BIOS call by VBIOS. The table below provides the standard panel ID table for three popular DTD panel modes.

Table 25. Example Panel ID Definitions

Panel ID	Mode
1	VGA - 640x480
2	SVGA - 800x600
3	XGA - 1024x768

8.3 Configuration Using User Build System (UBS)

8.3.1 Overview

The VBIOS UBS was developed to:

- Enable users to build a customized VBIOS binary. Customization includes features of the VBIOS, DTD timings, and addition and selection of AIM modules to the present code. This allows flexibility to the end user and on-site compilation to modify the VBIOS binary according to user needs.
- Enable compilation for the Intel® 845 and 85x chipsets.
- Automate modification of assembly sources based on customer input and automatically handle the compilation and linking of selected target object files to produce the output binaries.

The UBS is contained within the usr_bld folder that is part of the Intel® Embedded Graphics Drivers VBIOS release package. UBS consists of folders named 845 and 855, representing the supported hardware devices.

The Include folder contains files necessary to create a user-customized object file.

This file is later linked to produce the final binary that is placed in the relevant product folder according to specifications in the VBIOS customization file.

UBS consists of three main components:

- VBIOS launcher
- VBIOS configuration file (def_eg.txt)
- VBIOS customization tool

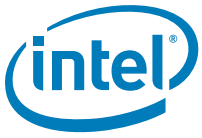
8.3.2 Requirements

- UBS must be installed on a system running a 32-bit Microsoft Windows operating system with the capability to execute DOS commands from a command line shell.

Note: When extracting the video BIOS package, make sure it is placed as close to the “root” as possible, as there is character limitation in DOS. For example:

Recommended: `C:\IEGD\usr_bld`
Not recommended: `C:\Documents and Settings\Administrator\My Documents\IEGD\usr_bld`

- The target machine must be installed with the open-source Watcom* C/C++ Version 11.0c compiler for DOS and MASM* 6.11 for DOS.
- The paths for the MASM assembler (ml) and the Watcom environments must be set up:
  - Go to <Control Panel><System><Advanced><Environment Variables><User Variables>, double-click on variable “Path”, and add the full path for `ml.exe` and `nmake.exe` in the <Variable Value> field.



8.3.3 VBIOS Launcher

1. Go to the USR_BLD folder and run go.bat. This is the main execution file.
2. Execute `go.bat` to create the release.

The go.bat file accepts a configuration file as an input argument. The default configuration file is `def_eg.txt`. An error is generated if this argument is not specified. This file is explained further in Section 8.3.4

User-selected configuration options are displayed when this file is executed, and any errors are indicated.

8.3.4 VBIOS Configuration File

The VBIOS configuration file contains customizable user options. A default example file called `def_eg.txt` is included along with the UBS. This file name has to be a command line parameter for go.bat. Multiple files can be created for different configurations as per user requirements. The file names must be DOS-compliant (maximum of eight characters). The VBIOS configuration file contains many options for user customization. A PCF file is used to configure DTD timings. The file `standard.pcf` contains an example of timing configurations.

This section explains the available options in the VBIOS Customization Tool (VCT) option configuration file (`def_eg.txt`). The VCT configuration file is divided into the following categories: VERSION, BUILD, GENERAL, PORT_DEVICE and BOOT. The sequence of these entries matters for correct implementation of the final exe when it is created. Each category has its own collection of options that can be changed. Do not leave any space for entries within a category. These categories are defined in the sections that follow.

8.3.4.1 Version Selection Category

This section is reserved and should not be modified from the default values in the `def_eg.txt` file distributed with the VBIOS.

8.3.4.2 Build Selection Category

The build selection category is used to select major build options. An example of this is shown in Figure 25 and subsequent sections give a detailed explanation of these options.

Figure 25. Build Settings

```
[BUILD]
HARDWARE= 845; 845,855
VGA=1 ; 1 to enable, 0 to disable
VESA=0; 1 to enable, 0 to disable
INTEL_5F=0; 1 to enable, 0 to disable
POST_DISPLAY_MSG=1; 1 to enable, 0 to disable
DISPLAY_STR_MSG="Intel EID Video BIOS"; not more than 60 chars
VESA_VBE_PM=1; 1 to enable, 0 to disable
```

8.3.4.2.1 Hardware

Only two types of hardware input are accepted. Acceptable parameters are:

- 845 for the Intel® 845GV chipset
- 855 for the Intel® 855GME or the Intel® 852GME chipsets

8.3.4.2.2 VGA

- Enables support for VGA functionality
- Enables selection of legacy VGA interface support
- Selecting VGA support ensures full compatibility with the documented standard IBM VGA BIOS, including standard VGA functions, register setting, mode resolutions, and RAM data area values
- Acceptable parameters are:
  - 1 to enable VGA support
  - 0 to disable VGA support

Note: This feature is hard-coded in the VBIOS to always be enabled.

8.3.4.2.3 VESA

- Enables support of VESA functionality
- VESA VBE standard allows software to set non-IBM* standard mode resolutions
- Acceptable parameters are:
  - 1 to enable VESA support
  - 0 to disable VESA support

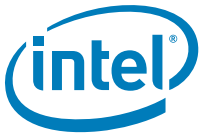
8.3.4.2.4 INTEL_5F

- BIOS extended interface functions
- Proprietary function calls to control operation of the extended features of the VBIOS
- Acceptable parameters are:
  - 1 to enable Intel_5f support
  - 0 to disable Intel_5f support

Note: Intel_5F settings are not supported in the current release.

8.3.4.2.5 POST_DISPLAY_MSG

- Enable POST display message
- Acceptable parameters are:
  - 1 to enable post display message support
  - 0 to disable post display message support



8.3.4.2.6 DISPLAY_STR_MSG

- String to display during POST
- This string is displayed only if POST_DISPLAY_MSG is enabled
- 60-character limit
- Enclose the message with open (“) and close (”) quotes

Note: This string will only be displayed during POST if POST_DISPLAY_MSG is enabled.

Note: An empty display string message is not advisable (e.g., " ").

8.3.4.2.7 VESA_VBE_PM

- Enables or disables power management support through function INT10 4F10.
- Acceptable parameters are:
  - 1 to enable power management support.
  - 0 to disable power management support.

8.3.4.3 General Selection Category

The general selection category shows the size and the DOS boot mode. An example is shown in Figure 26.

Figure 26. General Options

```
[GENERAL]
SIZE= 64; size in Kbytes
START_BOOT_MODE=0x50; DOS boot mode
```

8.3.4.3.1 Size

- Allows selection of VBIOS size.
- Acceptable parameter: 64 Kbytes.

8.3.4.3.2 START_BOOT_MODE

- Allows selection of required mode when DOS boots up.
- Acceptable parameters for VGA modes are:

Mode	Type	Resolution	Bits per Pixel
0x03	Text only	720x400	4
0x0F	Monotone	640x350	1
0x11	Graphics	640x480	2
0x12	Graphics	640x480	4
0x13	Graphics	320x200	8

Acceptable parameters for VESA modes are:

Mode	Resolution	Colors
0x30	640x480	256
0x32	800x600	256
0x34	1024x768	256
0x40	640x480	32 K
0x41	640x480	64 K
0x42	800x600	32 K
0x43	800x600	64 K
0x44	1024x768	32 K
0x45	1024x768	64 K
0x50	640x480	16 M
0x52	800x600	16 M
0x54	1024x768	16 M

Note: VESA must be enabled under the BUILD category to use the VESA modes.

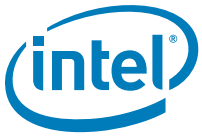
8.3.4.4 PORT_CONFIG Selection Category

The PORT_CONFIG selection category is used for Port and ADD configuration. An example is shown in Figure 27.

Note: EDID-less DTD timings customization is provided by a separate customization file. The default example is included in the `standard.pcf`. The `standard.pcf` file is parsed by the `pcf2iegd.exe` utility. After `go.bat` is executed, the DTD timings configuration file is automatically parsed to produce the necessary object file. Customization of DTD timings follow the EDID 1.0 standard. You can make multiple copies of the DTD timings file but must follow the format as shown in `standard.pcf`. The file can be renamed but must be specified in the DTD_TABLE settings in `def_eg.txt`. Only one file can be specified at a time.

Figure 27. Port Configuration Options

```
[PORT_CONFIG]
PORT_DEV_SUPPORT=1; 1 to enable, 0 to disable
ADDCARD_DOWNLOAD=1; 1 to enable, 0 to disable
REVERSE_DVO_COLOR_ORDER=0; 1 to enable, 0 to disable
PANEL_DETECT=0; 1 to enable, 0 to disable
DEFAULT_PANEL_ID=0; 0 to disable, any other number = panel ID
DEVICE_SELECT=0; 1 to enable, 0 to disable
;DTD_TABLE = "standard.pcf"
;STATIC_DEVICE0=C:\watcom\fal\si154.flx; user may insert as many as needed
;STATIC_DEVICE1=C:\watcom\falh164.flx; in sequence, with no blank lines
;STATIC_DEVICE2=binary_file_name3.xxx; in between entries. these client
;STATIC_DEVICE3=binary_file_name4.xxx; drivers get statically included
;STATIC_DEVICE4=binary_file_name5.xxx
```



8.3.4.4.1 PORT_DEV_SUPPORT

- Allows the selection of AIM device support
- Support for external TV-out digital encoder, DVI devices, and LVDS transmitter.
- If port device support is not selected, the AIM stub will not be linked in the execution file.
- Acceptable parameters are:
  - 1 to enable port device support
  - 0 to disable port device support

8.3.4.4.2 ADDCARD_DOWNLOAD

- This flag dictates whether or not ADD card AIM modules are downloaded and used.
- The parameter is either '1' for TRUE, meaning ADD card download of aim modules is supported, or '0' for FALSE, which means binaries from ADD card AIM modules are not downloaded and run.
- This does not mean that the ADD card DVO device cannot be used at all. Flex-AIM and Static-AIM modules that match the device ID of the ADD card DVO device can still be used.
- The purpose is to enable users to configure the VBIOS to use a flex or statically integrated AIM and not depend on the downloaded AIM module present on an available ADD card.

Note: The ADDCARD_DOWNLOAD setting is not supported in the current release.

8.3.4.4.3 REVERSE_DVO_COLOR_ORDER

- This flag allows the user to configure the VBIOS to flip the color order of the DVO signals.
- Causes data signals to be inverted from RGB (MSbs to LSbs) into the opposite direction.
- Parameters are:
  - 0 is default, ensuring normal operation.
  - 1 reverses the color order

8.3.4.4.4 PANEL_DETECT

Enables Panel ID detect BIOS function (5F40 through INT 15).

8.3.4.4.5 DEFAULT_PANEL_ID

- Assigns default Panel ID to VBIOS if:
  - PANEL_DETECT = 0 or
  - Panel ID detection through system BIOS is not successful.
- Settings DEFAULT_PANEL_ID = 0 means disabling this feature, as in no default panel ID is assigned to VBIOS.

8.3.4.4.6 DEVICE_SELECT

- Enables device selection through system BIOS
- Done through INT15h 5F35h.
- Only selected type of devices will be turned on if this feature is enabled.

Note: System BIOS must implement INT15 5F35h to enable user selection to be passed to VBIOS.

8.3.4.4.7 DTD_TABLE

- This entry contains a file name used to configure DTD timings.
- The default file included is `standard.pcf`.
- This file is parsed by the pcf2iegd utility to generate std_dtd.asm, which is then compiled to produce an object file specific to DTD timings. Eventually this file will be automatically linked to the final binary.
- File configuration (`standard.pcf`) follows standard EDID 1.0 documentation.

8.3.4.4.8 STATIC_DEVICES

- Allows selection of static AIM device support.
- Parameter consists of AIM static binary files.
- Copy these files to the specific intended product folder, or else copy the complete path to identify the files to be linked is required as a parameter.
- With the restriction of VBIOS size, any number of static binaries can be linked.
- Port_dev_support must be enabled for these static files to be linked with the final executable.

8.3.4.5 PORT_DEVICE Selection Category

The PORT_DEVICE selection category is used to configure AIM port device settings. With limitation to VBIOS size, any number of AIM port device settings can be included. Make sure to enable PORT_DEV_SUPPORT in the BUILD category to enable inclusion of these device settings in the final binary.

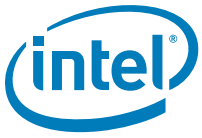
The configuration of the AIM port device settings dictates whether or not to activate available AIM port modules that are:

- Statically integrated in the VBIOS, called Static-AIM modules. Integration is also done through UBS3.
- Dynamically loaded Flex-AIM modules supported through MBI modules in system BIOS.

When ADDCARD_SUPPORT is enabled, the ADD card configuration always takes highest priority. The ADD card's AIM module is downloaded and activated. If any of the AIM port device settings entries (from this UBS configuration file) match the ADD card AIM settings, they are discarded, and the ADD card AIM settings would be used. The general rules are:

1. If ADDCARD_SUPPORT is TRUE:
   - — *And* an ADD card is present, it is downloaded and activated as first priority. Any AIM Port device settings in UBS that conflict with the AIM Port device ID or the ADD card are not used.
   - — *And* an ADD card is *not* present, the AIM Port device settings in UBS are used to search for static or Flex-AIM port device modules. Any found that match the device IDs are activated.
2. IF ADDCARD_SUPPORT is FALSE:
   - — The AIM Port device settings done in UBS are used to search for static or Flex-AIM port device modules. Any found that match the device IDs are activated.

In any case where the same Flex-AIM and Static-AIMs are available (same because of matching AIM device IDs), VBIOS uses the Static-AIM module.



An example is shown in Figure 28.

Figure 28. Port Device Options

```
[PORT_DEVICE]
Type=Port_EFP_TYPE      ; Port device type
ID=SI-164               ; String
Port=Port_DVO_C         ; Port usage = Port_DVO_A, DVO_B, etc
GPIO_BUS=MI2C           ; LTV, DDC, MI2C, etc
I2C_Address=0x70        ; address of i2c device
DDC_BUS=MDVI            ; LTV, DDC, MDVI, etc
Back_Light=1            ; 1 to enable, 0 to disable
```

8.3.4.5.1 Type

- Sets the device type.
- Parameters allowed are:
  - Port_TV_TYPE indicates TV encoder
  - Port_EFP_TYPE indicates TMDS connected to a DFP
  - Port_LFP_TYPE indicates LVDS encoder connected to FP.
  - Port_ADD_EFP_TYPE indicates an additional DFP.

8.3.4.5.2 ID

- Sets the device identification.
- Parameters allowed are string literals with length less than ten characters.
- Parameters are unique strings identifying the AIM module and are provided by the company that created the modules.
- The following table lists known device IDs.

Table 26. Device IDs

Device ID	Product
IN-LVDS	Internal LVDS on Intel® 852GME, 855GME, 915GV, 915GME chipsets
CH-7009-A	Chrontel 7009A/7009B/7011* as TV encoder
CH-7009-B	Chrontel 7009A/7009B/7301* as EFP encoder
NA-2501	National Semiconductor 2501* as LFP encoder
SI-164	Silicon Image 154/164* as EFP encoder
TH-164	THine 164* as EFP encoder

8.3.4.5.3 Port

- Sets the DVO port for the device.
- Parameters allowed are:
  - Port_DVO_B for all hardware configurations.
  - Port_DVO_C for all hardware configurations.

8.3.4.5.4 GPIO_BUS and DDC_BUS

- General-purpose I/O bus and DDC bus type settings.
- Parameters allowed are:
  - LCKCTRL for general I^2C
  - DDCA for analog CRT DDC
  - DDCP for DVI/LVDS DDC
  - MDDC for add card DDC
  - MI2C for add card I^2C
  - MDVI for add card I^2C or DDC

8.3.4.5.5 I2C_Address

- I^2C address.
- Parameter allowed: Hex value.

8.3.4.5.6 BACK_LIGHT

- Enables backlight support for current port through system BIOS.
- Done through INT15h 5F47h.
- Acceptable parameters are:
  - 1 to enable backlight support
  - 0 to disable backlight support

Note: System BIOS must implement INT15 5F47h to enable the backlight for panel selected.

8.3.4.6 BOOT Display Attachment Selection Category

- Allows selection of attach device and boot device
- Delimited by the equal sign; the left side indicates attached device and the right side indicates devices to boot
- Parameters allowed: CRT, TV1, EFP1, LFP, TV2, EFP2
  - TV1 - Indicates TV Out transmitter with single output
  - EFP1 - Indicates DVI transmitter with single output
  - TV2/EFP2 - Indicates transmitter with two external outputs (either TV Out or DVI). For example, Chrontel CH7009* is a DVI/TV Output Device. Depending on the AIM/PORT client module support, VBIOS can dictate which of these two signals to enable during boot. When TV function is selected, it is designated as TV2; when DVI function is selected, it should be set as EFP2.
- The VBIOS does not allow the combination of TV1 and TV2 to be turned on with a CRT, because display timings for both these display devices are so different that one would cause the other to be corrupted. Thus, the VBIOS only allows TV1 or TV2 to be turned on.
- An example is shown in Figure 29.

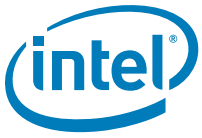


Figure 29. Boot Options

```
[Boot]
; Attached Dev Boot Dev
CRT=CRT
TV1=TV1
EFP1=EFP1
LFP,CRT=CRT
LFP,TV1=TV1
LFP,EFP1=EFP1
CRT,TV1=CRT
CRT,EFP1=CRT,EFP1
LFP,CRT,EFP1=CRT,EFP1
```

8.3.5 VBIOS Customization Tool (VCT)

The VCT contains a file called `vct.exe`. This tool is responsible for digesting the user-customizable file described above. It generates files called `config.asm` and `std_dtd.asm` (if the DTD option is selected) into a product-specific folder. The tool compiles this file to generate an object file. VCT then creates the target binary. The binary is located in the output folder of the hardware folder selected according to the requested hardware revision. The binary will be a RAM TSR (dosvga.exe) or a VGA VBIOS binary extractor (romvga.exe). VGA.bin is also automatically generated.

8.3.6 VBIOS Tips

1. During dosvga execution, to enable the static aim/port integration of AIM/PORT client modules, remember to:
   a. Run the UBS with the correct modifications in the Video Configuration File.
   b. Run "dosvga xxx.a yyy.b ... " where xxx.a and yyy.b are the aim/port client modules binaries that have been added in the Video Configuration File such as `si154.flx` and etc.
2. Make sure all the files in the `...\driver\usr_bld` folder are not set to read-only. If any of the files involved in the build process are set to read-only, the build process fails.
3. When UBS completes, if there was static integration of AIM/PORT client modules, the UBS merely runs "romvga xxx.a yyy.b ... " where xxx.a and yyy.b are the aim/port client modules binaries that have been added in the video configuration file such as `si154.flx`, etc. Thus, if romvga did not find the file in the proper folder, the user can rerun "romvga xxx.a yyy.b ... " and not have to repeat the entire UBS from go.bat. If VGA.BIN is not present, it indicates a failure in finding the specified file in STATIC_DEVICE entry at the VBIOS configuration file.

8.4 System BIOS Interface

The VBIOS is software that resides as an image in a ROM that is accessible by system BIOS during system boot time. The VBIOS image may be integrated with the system BIOS, which resides in the ROM on the motherboard, or the VBIOS may be a stand-alone image that resides in a ROM on a GC add-in board. In either case, the VBIOS image is never run in place, and some interfaces between VBIOS and system BIOS are required to improve flexibility.

8.5 VBIOS and Driver Compatibility

8.5.1 Data Dependencies Between VBIOS and Intel Graphics Drivers

The Intel Embedded Graphics Drivers do not depend on any data from the VBIOS, and will either use driver settings or select default values for the attached displays. This allows the driver to properly operate with incompatible BIOS or BIOS replacements.

The Intel Embedded Graphics Drivers will retrieve settings, such as panel ID and other display settings from the Embedded VBIOS. The Embedded VBIOS allows for configuration of display timings that can also be used for the Intel Embedded Graphics Drivers.

8.6 Video Modes

The VBIOS supports standard VGA modes. Table 27 lists the modes and vertical refresh rates that are supported by the VBIOS.

Note: Although IBM* labeled certain EGA modes with a (*) suffix and the VGA modes with a (+) suffix (such as mode 3, 3* and 3+), the VGA modes are so common that this document does not use the (+) suffix to refer to them.

The actual availability of any particular mode depends on the capabilities of the display device, the amount of memory installed, and other system parameters.

Table 27. Standard VGA Video Display Modes (Sheet 1 of 2)

Video Mode	Pixel Resolution	Color Depth	Mode Type	Display Adapter	Font Size	Character Resolution	Dot Clock (MHz)	Horiz. Freq. (KHz)	Vert Freq (Hz)	Video Memory (Kbytes)
00h	320 x 200	16 (gray)	Text	CGA	8 x 8	40 x 25	25	31.5	70	256
	360 x 400	16		VGA	9 x 16	40 x 25	28	31.5	70	256
01h	320 x 200	16	Text	CGA	8 x 8	40 x 25	25	31.5	70	256
	360 x 400	16		VGA	9 x 16	40 x 25	28	31.5	70	256
02h	640 x 200	16 (gray)	Text	CGA	8 x 8	80 x 25	25	31.5	70	256
	720 x 400	16		VGA	9 x 16	80 x 25	28	31.5	70	256
03h	640 x 200	16	Text	CGA	8 x 8	80 x 25	25	31.5	70	256
	720 x 400	16		VGA	9 x 16	80 x 25	28	31.5	70	256
04h	320 x 200	4	Graph	All	8 x 8	40 x 25	25	31.5	70	256
05h	320 x 200	4 (gray)	Graph	CGA	8 x 8	40 x 25	25	31.5	70	256
	320 x 200	4 (gray)		EGA	8 x 8	40 x 25	25	31.5	70	256
	320 x 200	4		VGA	8 x 8	40 x 25	25	31.5	70	256
06h	640 x 200	2	Graph	All	8 x 8	80 x 25	25	31.5	70	256
07h	720 x 350	Mono	Text	MDA	9 x 14	80 x 25	28	31.5	70	256
	720 x 350	Mono		EGA	9 x 14	80 x 25	28	31.5	70	256
	720 x 400	Mono		VGA	9 x 16	80 x 25	28	31.5	70	256
08h-0Ch	Reserved			-		-				
0Dh	320 x 200	16	Graph	E/VGA	8 x 8	40 x 25	25	31.5	70	256
0Eh	640 x 200	16	Graph	E/VGA	8 x 8	80 x 25	25	31.5	70	256

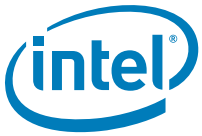


Table 27. Standard VGA Video Display Modes (Sheet 2 of 2)

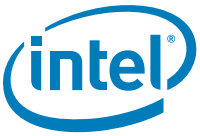
Video Mode	Pixel Resolution	Color Depth	Mode Type	Display Adapter	Font Size	Character Resolution	Dot Clock (MHz)	Horiz. Freq. (KHz)	Vert Freq (Hz)	Video Memory (Kbytes)
11h	640 x 480	2	Graph	VGA	8 x 16	80 x 30	25	31.5	60	256
12h	640 x 480	16	Graph	VGA	8 x 16	80 x 30	25	31.5	60	256
13h	320 x 200	256	Graph	VGA	8 x 8	40 x 25	25	31.5	70	256

Table 28 lists VESA modes supported by the Legacy VBIOS.

Table 28. VESA Modes Supported by Legacy VBIOS

Mode	VBE Mode	Resolution and bpp (bits per pixel)	Colors	Type	Dot Clock (MHz)	Frequency (Hz)	Video Memory (Mbytes)
30h	101h	640x480x8	256	Graph	25.175	60	0.29
32h	103h	800x600x8	256	Graph	40	60	0.46
34h	105h	1024x768x8	256	Graph	65	60	0.75
38h	107h	1280x1024x8	256	Graph	108	60	1.25
3Ah	13Ah	1600x1200x8	256	Graph	162	60	1.83
3Ch	13Ch	1920x1440x8	256	Graph	234	60	2.77
41h	111h	640x480x16	64K	Graph	25.175	60	0.59
43h	114h	800x600x16	64K	Graph	40	60	0.92
45h	117h	1024x768x16	64K	Graph	65	60	1.50
49h	11Ah	1280x1024x16	64K	Graph	108	60	2.5
4Bh	14Bh	1600x1200x16	64K	Graph	162	60	3.67
4Dh	14Dh	1920x1440x16	64K	Graph	234	60	5.5
50h	112h	640x480x32	16M	Graph	25.175	60	1.17
52h	115h	800x600x32	16M	Graph	40	60	1.83
54h	118h	1024x768x32	16M	Graph	65	60	3.0
58h	11Bh	1280x1024x32	16M	Graph	108	60	5.0
5Ah	15Ah	1600x1200x32	16M	Graph	162	60	7.32

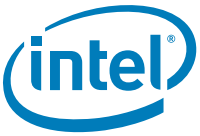
This page is intentionally left blank.



Appendix A Example INF File

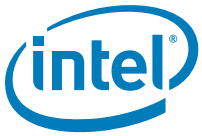
```
;*****************************************************************************
; Filename: iegd.inf
;
; Copyright (c) 2007 Intel Corporation.  All rights reserved.
;
;*****************************************************************************

[Version]
Signature="$WINDOWS NT$"
Class=Display
ClassGUID={4D36E968-E325-11CE-BFC1-08002BE10318}
Provider=%Intel%
CatalogFile=iegd.cat
DriverVer = 08/18/2007,8.0
;=============================================================================
[SourceDisksNames]
1=%DiskDesc%,,,""

[SourceDisksFiles]
iegdmini.sys = 1
iegddis.dll  = 1
iegd3dg3.dll = 1
iegd3dg4.dll = 1
ch7009.sys   = 1
ch7017.sys   = 1
fs454.sys    = 1
lvds.sys     = 1
ns2501.sys   = 1
ns387.sys    = 1
sii164.sys   = 1
ti410.sys    = 1
th164.sys    = 1
sdvo.sys     = 1
tv.sys       = 1


;=============================================================================
[DestinationDirs]
DefaultDestDir     = 11; System directory
iegd.Display_alm  = 11
iegd.Display_nap  = 11
iegd.Display_gn4  = 11
iegd.Miniport     = 12; Drivers directory
iegd.PortDrvs_alm = 12
iegd.PortDrvs_nap = 12
iegd.PortDrvs_gn4 = 12

;=============================================================================
[Manufacturer]
%Intel%=Intel.Mfg

;=============================================================================
[Intel.Mfg]
%Intel% %i830m%   = iegd_alm, PCI\VEN_8086&DEV_3577
%Intel% %i835%    = iegd_alm, PCI\VEN_8086&DEV_357B
```

```
%Intel% %i845%    = iegd_alm, PCI\VEN_8086&DEV_2562
%Intel% %i855%    = iegd_alm, PCI\VEN_8086&DEV_3582
%Intel% %i865%    = iegd_alm, PCI\VEN_8086&DEV_2572

%Intel% %i915GD0% = iegd_nap, PCI\VEN_8086&DEV_2582
%Intel% %i915GD1% = iegd_nap, PCI\VEN_8086&DEV_2782
%Intel% %i915AL0% = iegd_nap, PCI\VEN_8086&DEV_2592
%Intel% %i915AL1% = iegd_nap, PCI\VEN_8086&DEV_2792
%Intel% %i945LP0% = iegd_nap, PCI\VEN_8086&DEV_2772
%Intel% %i945LP1% = iegd_nap, PCI\VEN_8086&DEV_2776
%Intel% %i945CT0% = iegd_nap, PCI\VEN_8086&DEV_27A2
%Intel% %i945CT1% = iegd_nap, PCI\VEN_8086&DEV_27A6

%Intel% %i965BW0% = iegd_gn4, PCI\VEN_8086&DEV_2982
%Intel% %i965BW1% = iegd_gn4, PCI\VEN_8086&DEV_2983
%Intel% %iG9650%  = iegd_gn4, PCI\VEN_8086&DEV_29A2
%Intel% %iG9651%  = iegd_gn4, PCI\VEN_8086&DEV_29A3
%Intel% %iQ9650%  = iegd_gn4, PCI\VEN_8086&DEV_2992
%Intel% %iQ9651%  = iegd_gn4, PCI\VEN_8086&DEV_2993
%Intel% %i946GZ0% = iegd_gn4, PCI\VEN_8086&DEV_2972
%Intel% %i946GZ1% = iegd_gn4, PCI\VEN_8086&DEV_2973
%Intel% %i965GM0% = iegd_gn4, PCI\VEN_8086&DEV_2A02
%Intel% %i965GM1% = iegd_gn4, PCI\VEN_8086&DEV_2A03

;=============================================================================
[iegd_alm.GeneralConfigData]
MaximumNumberOfDevices = 2
MaximumDeviceMemoryConfiguration = 128

[iegd_nap.GeneralConfigData]
MaximumNumberOfDevices = 2
MaximumDeviceMemoryConfiguration = 256

[iegd_gn4.GeneralConfigData]
MaximumNumberOfDevices = 2
MaximumDeviceMemoryConfiguration = 512

;=============================================================================
[iegd_alm]
CopyFiles = iegd.Miniport, iegd.Display_alm, iegd.PortDrvs_alm

[iegd_nap]
CopyFiles = iegd.Miniport, iegd.Display_nap, iegd.PortDrvs_nap

[iegd_gn4]
CopyFiles = iegd.Miniport, iegd.Display_gn4, iegd.PortDrvs_gn4

;=============================================================================
[iegd.Miniport]
iegdmini.sys

[iegd.Display_alm]
iegddis.dll
iegd3dg3.dll

[iegd.Display_nap]
iegddis.dll
iegd3dg3.dll

[iegd.Display_gn4]
iegddis.dll
iegd3dg4.dll

[iegd.PortDrvs_alm]
ch7009.sys
ch7017.sys
fs454.sys
```



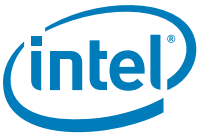
```
lvds.sys
ns2501.sys
ns387.sys
sii164.sys
ti410.sys
th164.sys

[iegd.PortDrvs_nap]
sdvo.sys
lvds.sys
tv.sys

[iegd.PortDrvs_gn4]
sdvo.sys
lvds.sys

;=============================================================================
[iegd_alm.Services]
AddService = iegdmini, 0x00000002, iegd_Service_Inst, iegd_EventLog_Inst
AddService = ch7009, ,ch7009_Service_Inst, iegd_EventLog_Inst
AddService = ch7017, ,ch7017_Service_Inst, iegd_EventLog_Inst
AddService = fs454,  ,fs454_Service_Inst, iegd_EventLog_Inst
AddService = lvds,   ,lvds_Service_Inst, iegd_EventLog_Inst
AddService = ns2501, ,ns2501_Service_Inst, iegd_EventLog_Inst
AddService = ns387,  ,ns387_Service_Inst, iegd_EventLog_Inst
AddService = sii164, ,sii164_Service_Inst, iegd_EventLog_Inst
AddService = ti410,  ,ti410_Service_Inst, iegd_EventLog_Inst
AddService = th164,  ,th164_Service_Inst, iegd_EventLog_Inst

[iegd_nap.Services]
AddService = iegdmini, 0x00000002, iegd_Service_Inst, iegd_EventLog_Inst
AddService = lvds,   ,lvds_Service_Inst, iegd_EventLog_Inst
AddService = sdvo,   ,sdvo_Service_Inst, iegd_EventLog_Inst
AddService = tv,     ,tv_Service_Inst, iegd_EventLog_Inst

[iegd_gn4.Services]
AddService = iegdmini, 0x00000002, iegd_Service_Inst, iegd_EventLog_Inst
AddService = lvds,   ,lvds_Service_Inst, iegd_EventLog_Inst
AddService = sdvo,   ,sdvo_Service_Inst, iegd_EventLog_Inst

;=============================================================================
[iegd_Service_Inst]
ServiceType    = 1
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = 0
LoadOrderGroup = Video
ServiceBinary  = %12%\iegdmini.sys

[ch7009_Service_Inst]
DisplayName    = “ch7009”
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\ch7009.sys

[ch7017_Service_Inst]
DisplayName    = “ch7017”
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\ch7017.sys

[fs454_Service_Inst]
DisplayName    = “fs454”
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
```

```
ServiceBinary  = %12%\fs454.sys

[lvds_Service_Inst]
DisplayName    = “lvds”
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\lvds.sys

[ns2501_Service_Inst]
DisplayName    = “ns2501”
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\ns2501.sys

[ns387_Service_Inst]
DisplayName    = “ns387”
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\ns387.sys

[sii164_Service_Inst]
DisplayName    = “sii164”
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\sii164.sys

[ti410_Service_Inst]
DisplayName    = “ti410”
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\ti410.sys


[th164_Service_Inst]
DisplayName    = “th164”
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\th164.sys

[sdvo_Service_Inst]
DisplayName    = “sdvo”
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\sdvo.sys

[tv_Service_Inst]
DisplayName    = “tv”
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\tv.sys

;=================================================================================
[iegd_EventLog_Inst]
AddReg = iegd_EventLog_AddReg

[iegd_EventLog_AddReg]
HKR,,EventMessageFile,0x00020000,”%SystemRoot%\System32\IoLogMsg.dll;%SystemRoot%\Syste
m32\drivers\iegdmini.sys”
HKR,,TypesSupported,0x00010001,7
```



```
;=============================================================================
[iegd_alm.SoftwareSettings]
AddReg = iegd_SoftwareDeviceSettings_alm

[iegd_nap.SoftwareSettings]
AddReg = iegd_SoftwareDeviceSettings_nap

[iegd_gn4.SoftwareSettings]
AddReg = iegd_SoftwareDeviceSettings_gn4

;=============================================================================
[iegd_SoftwareDeviceSettings_alm]
HKR,, InstalledDisplayDrivers, %REG_MULTI_SZ%, iegddis
HKR,, MultiFunctionSupported, %REG_MULTI_SZ%, 1
HKR,, VgaCompatible, %REG_DWORD%, 0
HKR,, PcfVersion,    %REG_DWORD%, 0x0700

HKR,, No_D3D, %REG_DWORD%, 0
HKR,, PortDrivers, %REG_SZ%, “ch7009 ch7017 fs454 lvds ns2501 ns387 sii164 ti410 th164”

;-----------------------------------------------------------------------------
[iegd_SoftwareDeviceSettings_nap]
HKR,, InstalledDisplayDrivers, %REG_MULTI_SZ%, iegddis
HKR,, MultiFunctionSupported, %REG_MULTI_SZ%, 1
HKR,, VgaCompatible, %REG_DWORD%, 0
HKR,, PcfVersion,    %REG_DWORD%, 0x0700

HKR,, No_D3D, %REG_DWORD%, 0

HKR,, PortDrivers, %REG_SZ%, “sdvo lvds tv”

;-----------------------------------------------------------------------------
[iegd_SoftwareDeviceSettings_gn4]
HKR,, InstalledDisplayDrivers, %REG_MULTI_SZ%, iegddis
HKR,, MultiFunctionSupported, %REG_MULTI_SZ%, 1
HKR,, VgaCompatible, %REG_DWORD%, 0
HKR,, PcfVersion,    %REG_DWORD%, 0x0700

HKR,, No_D3D, %REG_DWORD%, 0

HKR,, PortDrivers, %REG_SZ%, “sdvo lvds”

;=============================================================================
[Strings]

;-----------------------------------------------------------------------
; Localizable Strings
;-----------------------------------------------------------------------
Intel=”Intel Corporation”
DiskDesc=”Embedded Installation”

Intel="Intel Corporation"
DiskDesc="Embedded Installation"

i830m="830M Embedded Graphics Controller"
i835="835 Embedded Graphics Controller"
i845="845 Embedded Graphics Controller"
i855="855 Embedded Graphics Controller"
i865="865 Embedded Graphics Controller"

i915GD0="915G/915GV/910GL Embedded Graphics Controller Function 0"
i915GD1="915G/915GV/910GL Embedded Graphics Controller Function 1"
i915AL0="915GM/915GMS/915GME/910GML/910GMLE Embedded Graphics Controller Function 0"
i915AL1="915GM/915GMS/915GME/910GML/910GMLE Embedded Graphics Controller Function 1"
i945LP0="945G Embedded Graphics Controller Function 0"
i945LP1="945G Embedded Graphics Controller Function 1"
```

```
i945CT0="945GM Embedded Graphics Controller Function 0"
i945CT1="945GM Embedded Graphics Controller Function 1"
i945WB0="945GME Embedded Graphics Controller Function 0"
i35BL0="Q35 Embedded Graphics Controller Function 0"
i35BL1="Q35 Embedded Graphics Controller Function 1"

i965BW0="965G Embedded Graphics Controller Function 0"
i965BW1="965G Embedded Graphics Controller Function 1"
iG9650="G965 Embedded Graphics Controller Function 0"
iG9651="G965 Embedded Graphics Controller Function 1"
iQ9650="Q963/Q965 Embedded Graphics Controller Function 0"
iQ9651="Q963/Q965 Embedded Graphics Controller Function 1"
i946GZ0="946GZ Embedded Graphics Controller Function 0"
i946GZ1="946GZ Embedded Graphics Controller Function 1"
i965GM0="GM965 Embedded Graphics Controller Function 0"
i965GM1="GM965 Embedded Graphics Controller Function 1"
i965GME0="GME965 Embedded Graphics Controller Function 0"
i965GME1="GME965 Embedded Graphics Controller Function 1"

;-------------------------------------------------------------------------
; Non Localizable Strings
;-------------------------------------------------------------------------
SERVICE_BOOT_START     = 0x0
SERVICE_SYSTEM_START   = 0x1
SERVICE_AUTO_START     = 0x2
SERVICE_DEMAND_START   = 0x3
SERVICE_DISABLED       = 0x4

SERVICE_KERNEL_DRIVER  = 0x1

SERVICE_ERROR_IGNORE   = 0x0; Continue on driver load fail
SERVICE_ERROR_NORMAL   = 0x1; Display warn, but continue
SERVICE_ERROR_SEVERE   = 0x2; Attempt LastKnownGood
SERVICE_ERROR_CRITICAL = 0x3; Attempt LastKnownGood, BugCheck

REG_EXPAND_SZ = 0x00020000
REG_MULTI_SZ  = 0x00010000
REG_DWORD     = 0x00010001
REG_SZ        = 0x00000000
```



Appendix B Port Driver Attributes

B.1 Standard Port Driver Attributes

Port drivers are modules within the IEGD driver suite that control GMCH-specific modules such as GMCH LVDS, GMCH TV or add-on modules to GMCH. Table 29 lists the attributes available to port drivers. Some of these standard attributes can be customized for specific port drivers and are detailed in the following sections of this appendix.

Note: Not all standard attributes are supported by all port drivers. Please see the following sections for details on the specific attributes supported by each port driver. Flat panel settings are specified via the FPINFO options of the configuration; please see Table 10 in Section 3.0.

Table 29. Standard Port Driver Attributes

Attribute Name	Attribute ID Number	Description
BRIGHTNESS	0	Brightness adjustment.
CONTRAST	1	Contrast adjustment.
HUE	2	Hue adjustment.
FLICKER	3	Setting to reduce flicker.
HPOSITION	4	Controls the horizontal position of the display.
VPOSITION	5	Controls the vertical position of the display.
HSCALE	6	Horizontal scaling ratio.
VSCALE	7	Vertical scaling ratio.
TVFORMAT	8	TV formats are device-specific.
DISPLAY TYPE	9	Allows selection of different displays for multi-display devices. This attribute is device-specific.
LUMA FILTER	10	TV Luma Filter adjustment.
CHROMA FILTER	11	ChromaFilter adjustment.
TEXT FILTER	12	Text Filter adjustment.
TV OUTPUT TYPE	14	TV output types. This attribute is device-specific.
SATURATION	15	Saturation adjustment.
PANEL FIT	18	Panel fitting. Yes or no.
SCALING RATIO	19	Output Scaling. Device-specific.
FP BACKLIGHT ENABLE	20	Enable flat panel backlight.
PANEL DEPTH	26	Can be either 18 or 24. 18 specifies 6-bit output per color, 24 specifies 8-bit output per color.
DUAL CHANNEL PANEL	27	Is it a dual channel panel or not? Takes 0 or 1.
GANG MODE	28	For achieving a Gang mode output using two digital ports.

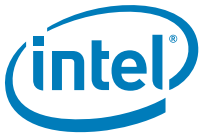
Table 29. Standard Port Driver Attributes (Continued)

Attribute Name	Attribute ID Number	Description
GANG MODE EVEN ODD	29	Gang display even or odd. This attribute is to be set along with Gang mode (28). This mode (Gang Mode Even Odd) puts even pixels on one digital port and odd pixels on the other, and needs to be selected based on the display panel used.
REVERSE DVO DATA	30	Reverses DVO data order.
SHARPNESS	31	Sharpness.
HWCONFIG	32	Hardware Configuration for DVO encoders that support multiple configurations.
HORZFILTER	33	Horizontal Filter.
VERTFILTER	34	Vertical Filter.
FRAME BUFFER GAMMA	35	Framebuffer gamma correction.
FRAME BUFFER BRIGHTNESS	36	Framebuffer brightness.
FRAME BUFFER CONTRAST	37	Framebuffer contrast.
2D FLICKER	39	Two-dimension flicker.
ADAPTIVE FLICKER	40	Adaptive flicker.
HORIZONTAL OVERSCAN	41	Horizontal overscan.
VERTICAL OVERSCAN	42	Vertical overscan.
SPREAD SPECTRUM CLOCKING	43	Spectrum Clocking
DOT_CRAWL	44	Dot crawl affects the edges of color and manifests itself as moving dots of color along these edges.
DITHER	45	Dither setting
PANEL PROTECT HSYNC	46	Horizontal sync panel protection
PANEL PROTECT VSYNC	47	Vertical sync panel protection
PANEL PROTECT PIXCLK	48	Pixel clock protection
LVDS PANEL TYPE	49	This is used to select SPWG vs. OpenLDI panel types. 0-SPWG 1-OpenLDI.
GANG DVO CLOCK INVERSION	56	Controls DVO clock inversion in Gang mode. This attribute is to be set along with the "Gang Mode" attribute (28).
VGA 2X IMAGE	57	Controls VGA image in Gang mode.
TEXT ENHANCEMENT	58	Controls text tuning.
MAINTAIN ASPECT RATIO	59	This controls scaled image to match source image aspect ratio or full screen image.
FIXED TIMING	60	This indicates whether the attached display is a fixed timing display.

B.2 Port Driver Attributes

This section provides the supported attributes for each of the port drivers.

Note: In the following tables, device-specific (non-standard) attributes are highlighted in gray.



B.2.1 Internal LVDS Port Driver Attributes (Mobile chipsets only)

Table 30. Internal LVDS Port Driver Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
PANELDEPTH	26	Specify Panel Depth based on connected panel.	Default is 18, however, on some GMCH chipsets 24-bit also is supported. For example, GM965 supports both 18 and 24-bit outputs.
DUALCHANNEL	27	Single or Dual Channel Panel	0 = Single 1 = Dual Default is 0.
DITHER	45	On and off Dithering	0 = on 1 = off Default: • dither on for 18-bit panels • dither off for 24-bit panels.
FIXED TIMING	60	This indicates whether attached display is a fixed timing display.	0 = on 1 = off
DATA FORMAT	32769	Panel data format.	Default is 0. Format 0 is supported in 18-bit panel depth. Format 1 is supported in 24-bit panel depth.

B.2.2 CRT (Analog) Port Driver Attributes

Note: The analog port driver is included in the driver by default, unlike other port drivers available for selection as part of the driver configuration.

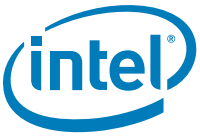
Table 31. CRT (Analog) Port Driver Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
FIXED TIMING	60	Set this attribute if the attached display supports only one timing.	0 = Not a fixed timing display. 1 = Fixed timing display. Default is 0.

B.2.3 Internal TV Out Port Driver Attributes (Mobile chipsets only)

Table 32. Internal TV Out Port Driver Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
BRIGHTNESS	0	Screen brightness	0-100. Default is 50.
CONTRAST	1	Color contrast	0-7. Default is 3.
HUE	2	Hue adjustment	0-100. Default is 0.
TV FLICK FILTER	3	TV Flicker Filter. The higher the value, the higher the amount of flicker filtering and text enhancement.	0-1000. Default is 999.
H POSITION	4	Horizontal Position. Increasing the value moves the image to the right and decreasing the value moves the image to the left.	0-511. Default is 64.
V POSITION	5	Vertical Position. The value represents the TV line number relative to the VGA vertical sync. Increasing the value moves the image down and decreasing the value moves the image up.	0-511. Default is 0.
TV FORMAT	8	TV formats are device-specific.	Default is NTSC-M (1).
TV OUTPUT	14	TV output types. This attribute is device-specific. ***Note:*** TV output types are limited to S-Video and Composite for the VBIOS.	Default is S-VIDEO (2).
OVERSCAN/SCALING RATIO	19	Output Scaling.	0-1000. Default is 350.



B.2.4 Chrontel CH7009/CH7010 Port Driver TV Attributes

Table 33 lists the TV attributes for the Chrontel CH7009/CH7010 port drivers.

Note: For flat panel backlight timing settings, please see Table 10 in Section 3.0.

Table 33. Chrontel CH7009/CH7010 Port Driver TV Attributes (Sheet 1 of 2)

Attribute Name	Attribute ID	Description	Possible Ranges
BRIGHTNESS	0	Screen brightness	0-100. Default is 50. Recommend 61 for NTSC-M, NTSC-443/433, and PAL-M. Recommend 37 for NTSC-J. Recommend 42 for PAL-B/D/G/H/I/N formats.
CONTRAST	1	Color contrast. A setting of 0 reduces the contrast, a setting of 1 leaves the image unchanged, and values beyond 1 increase the contrast.	0-7. Default is 3.
FLICKERFILTER	3	TV Flicker Filter. The higher the value, the higher the amount of flicker filtering and text enhancement.	0-1000. Default is 999. Recommend 999 or 1000.
HPOSITION	4	Horizontal Position. Increasing the value moves the image to the right and decreasing the value moves the image to the left.	0-511. Default is 64.
VPOSITION	5	Vertical Position. The value represents the TV line number relative to the VGA vertical sync. Increasing the value moves the image down and decreasing the value moves the image up.	0-511. Default is 0.
TVFORMAT	8	TV Format	1 = NTSC-M 2 = NTSC-M-J 3 = NTSC-433 4= PAL-M 5= PAL-B 6= PAL-G 7= PAL-D 8= PAL-H 9= PAL-I 10= PAL-N 11 = PAL-60 The default is 1, NTSC-M.
DISPLAY TYPE	9	Attached display type.	1 = DVI Flat panel 2 = TV It defaults to whatever display is connected. If both DVI and TV are connected or no display is connected, then it defaults to DVI. Default behavior can be overridden by setting this attribute to the required value. This is a useful attribute to set if the driver isn't detecting the correct display properly or no display is connected.

Table 33. Chrontel CH7009/CH7010 Port Driver TV Attributes (Continued) (Sheet 2 of 2)

Attribute Name	Attribute ID	Description	Possible Ranges
LUMA_FILTER	10	TV Luma Filter used in the scaling and flicker reduction block applied to low and high frequency of the luminance signal according to their specific scaling ratio.	0-3. Default is 3.
CHROMA_FILTER	11	TV Chroma Filter. This value limits the bandwidth of the chroma signal in the CVBS and S-Video output signals. (Please see Chrontel CH7009 datasheets/specifications for a description of the Video Bandwith register.)	0 = less bandwidth 1= more bandwidth. Default is 1.
TEXT_FILTER	12	TV Text Filter that controls the text enhancement capability designed in the chip. A value of 0 minimizes the enhancement feature while a value of 7 maximizes it.	0-7. Default is 3.
TV OUTPUT TYPE	14	TV Output Type	1 = COMPOSITE 2 = S-VIDEO 3 = CPSTSV (Composite and S-VIDEO) 4 = SCART1 5 = SCART2 Note: TV Output Types are limited to S-Video and Composite for the VBIOS. These only apply to TV mode, not Flat Panel mode (see attr 9).
SCALING_RATIO	19	Overscan	0 = Normal 1 = Overscan
VGA_BYPASS	32769	VGA Bypass. This attribute is associated with the Hardware Configuration attribute.	0 = disable VGA Bypass. 1 = enable VGA Bypass Default is 0, disable VGA Bypass.
DOT_CRAWL	32770	TV Dot Crawl (NTSC only). Dot crawl affects the edges of color and manifests itself as moving dots of color along these edges. A value of 1 freezes dot crawl; 0 allows dot crawl to run freely.	0 = have Dot Crawl run freely 1 = freeze Dot Crawl Default is 0.
HW_CONFIG	32771	Hardware Configuration. This attribute works in conjunction with the VGA Bypass and Display Device attributes.	1-2. Default is 1. If set to 1 and: VGA Bypass = 1 — Both DVI and VGA are enabled VGA Bypass = 0 — DVI or TV are enabled If set to 2 and: DVI connection detected — only DVI is enabled no DVI connection detected and VGA Bypass = 1 — only VGA Bypass enabled no DVI connection detected and VGA Bypass = 0 — only DVI enabled.

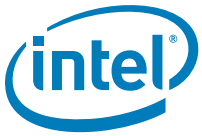


Table 34. CH7009 DVI Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
DISPLAY TYPE	9	Attached display type.	1 = DVI Flat panel 2 = TV It defaults to whatever display is connected. If both DVI and TV are connected or no display connected, then it defaults to DVI. Default behavior can be overridden by setting this attribute to the required value. This is a useful attribute to set if the driver isn't detecting the correct display properly or no display is connected.
FIXED TIMING	60	This indicates whether the attached display is a fixed timing display.	0 = on 1 = off
VGAOUT	32769	Enables or Disables VGA Output, which controls the VGA Bypass associated with the 32771 attribute.	0 = Disable 1 = Enable Default is 0.
HW_CONFIG	32771	Hardware Configuration. This attribute works in conjunction with the VGA Bypass and Display Device attributes.	1-2. Default is 1. If set to 1 and: VGA Bypass = 1 — Both DVI and VGA are enabled VGA Bypass = 0 — DVI or TV are enabled If set to 2 and: DVI connection detected — only DVI is enabled no DVI connection detected and VGA Bypass = 1 — only VGA Bypass enabled no DVI connection detected and VGA Bypass = 0 — only DVI enabled.

B.2.5 Chrontel CH7017/CH7305 Port Driver Attributes

Table 35 shows the attributes for the Chrontel CH7017 and CH7305 port drivers.

Note: For flat panel backlight timing settings, please see Table 10 in Section 3.0.

Table 35. Chrontel CH7017/CH7305 TV Attributes (Sheet 1 of 2)

Attribute Name	Attribute ID	Description	Possible Ranges
BRIGHTNESS	0	Screen brightness	0-100. Default is 50. Recommend 61 for NTSC-M, NTSC-443/433, and PAL-M. Recommend 37 for NTSC-J. Recommend 42 for PAL-B/D/G/H/I/N formats.
CONTRAST	1	Color contrast	0-7. Default is 3.
FLICKERFILTER	3	TV Flick Filter	0-1000. Default is 999. Recommend 999 or 1000.
HPOSITION	4	Horizontal Position. This shifts the displayed TV image in horizontally to center the image on the screen. Increasing the value moves the image to the right and decreasing the value moves the image to the left.	0-512. Default is 46. Increment by 1.
VPOSITION	5	Vertical Position. This shifts the displayed TV image in a vertically to center the image. The value represents the TV line number relative to the VGA vertical sync. Increasing the value delays the output of TV vertical sync, thus moving the image down. Decreasing the value moves the image up.	0-512. Default is 6. Increment by 1.
TVFORMAT	8	TV Format	1 = NTSC-M 2 = NTSC-M-J 3 = NTSC-433 4= PAL-M 5= PAL-B 6= PAL-G 7= PAL-D 8= PAL-H 9= PAL-I 10= PAL-N 11 = PAL-60
DISPLAY TYPE	9	Attached display type.	1 = LVDS Flat panel 2 = TV It defaults to whatever display is connected. If both LVDS and TV is connected or no display is connected, then it defaults to LVDS. Default behavior can be overridden by setting this attribute to the required value. This is a useful attribute to set if the driver isn't detecting the correct display properly or no display is connected.

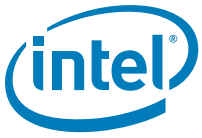


Table 35. Chrontel CH7017/CH7305 TV Attributes (Sheet 2 of 2)

Attribute Name	Attribute ID	Description	Possible Ranges
LUMA_FILTER	10	TV Luma Filter	0-3. Default is 3.
CHROMAFILTER	11	TV Chroma Filter. This value limits the bandwidth of the chroma signal in the CVBS and S-Video output signals. (Please see Chrontel CH7009 datasheet/specifications for a description of the Video Bandwidth register.)	0 = less bandwidth 1= more bandwidth. Default is 1.
TEXTFILTER	12	TV Text Filter	0-7. Default is 3.
TVOUT_TYPE	14	TV Output Types	1 = COMPOSITE 2 = S-VIDEO 3 = COMPONENT 4 = CPSTSV (Composite and S-Video) 5 = RGB 6 = RGB+CVBS 7 = SCART1 8 = SCART2 ***Note:*** TV Output Types are limited to S-Video and Composite for the VBIOS.
SCALINGRATIO	19	Screen scale ratio	1 = 5 to 4 2 = 1 to 1 3 = 7 to 8 (NTSC only) 4 = 5 to 6 5 = 5 to 7 (PAL only) 6 = 5 to 8 7 = 5 to 9 8 = 3 to 4 (NTSC only) 9 = 7 to 10 (NTSC only) 10 = 1 to 2 (NTSC only)

Table 36. Chrontel CH7017/CH7305 LVDS Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
PANELDEPTH	26	Panel Depth defines the picture format, which is either 18 bits or 24 bits.	18 or 24. Default is 18. For flat panels, refer to the panel specification.
SINGLE/DUALCHANNEL	27	Single/Dual Channel defines the chip channel mode.	0 = Single 1 = Dual Default is 0. For flat panels, refer to the panel specification.
DITHER	45	Dither setting	0 = Disable 1 = Enable Default = 0.
LVDS PANEL TYPE	49	Opens LDI or SPWG. This determines the encoded data format.	0 = SPWG 1 = OpenLDI Default is 1.
TEXTENHANCEMENT	58	Determines how text is rendered on a display. (Supported only in full-mode.)	0 = Smooth 1 = Normal 2 = Plain 3 = Sharp 4 = Very sharp Default is 1, Normal.
FIXED TIMING	60	This indicates whether attached display is a fixed timing display.	0 = on 1 = off
VGAOUT	32769	Enables or Disables VGA Output, which controls the VGA Bypass associated with the 32771 attribute.	0 = Disable 1 = Enable Default is 0.
DATAPATH	32770	Data Path controls the data path from D1 and D2 input to the internal TV and LVDS blocks.	1 = routes D1 to both blocks 2 = routes D1 to LVDS and D2 to TV (DVOB) 3 = routes D1 to TV and D2 to LVDS (DV0C) 4 = routes D2 to both blocks Default is 3.
DOT_CRAWL	32772	Dot Crawl affects the edges of color and manifests itself as moving dots of color along these edges. A setting of 1 freezes dot crawl while a setting of 0 allows dot crawl to run freely. (NTSC only)	0 = Have Dot Crawl run freely 1 = Freeze Dot Crawl Default is 0.
EMICOUPLINGCAP	32776	EMI Coupling CAP (EMI/PLL Setting). This controls the LVDS PLL Capacitor, which allows coupling of any signal to the on chip loop filter capacitor. Note: EMI settings are subject to change upon panel specifications.	1-15. Increment by 1. 1 for 1024x768 panel 9 for 1400x1050 panel 10 for 1600x1200 panel

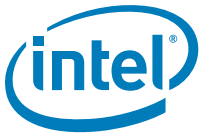


Table 36. Chrontel CH7017/CH7305 LVDS Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
EMIFDDIVIDER	32777	The EMI FD Divider defines the LVDS PLL Spread Spectrum Frequency Divider Control.	0-2048 If input clock frequency is: 40, enter 51 65, enter 127 108, enter 150 162, enter 290
EMIFBDIVIDER	32778	The EMI FB Divider defines the LVDS PLL Spread Spectrum Frequency Feedback Divider Control.	0-2048 Increment by 1. Default = 7.
EMILOOPFILTERRESISTOR	32779	The EMI Loop Filter Resistor controls the LVDS PLL Loop Filter Resistor.	Range from 0-7 0 = 1800 ohms 1 = 2600 ohms 2 = 1000 ohms 3 = 2300 ohms 4 = 21,800 ohms 5 = 42,600 ohms 6 = 11,000 ohms 7 = 73,200 ohms Default is 2. Recommend 3 if IEGD 3.3 driver or earlier is installed.
EMIREDUCTIONRESISTOR	32780	The EMI Reduction Resistor determines the Spread Spectrum Oscillator Frequency setting or the amplitude of an external frequency source.	Range from 0-7 0 = 0 µA 1 = 10 µA 2 = 20 µA 3 = 30 µA 4 = 40 µA 5 = 50 µA 6 = 60 µA 7 = 70 µA Default is 3. Recommend 4 if IEGD 3.3 driver or earlier is installed.

B.2.6 Chrontel CH7307 Port Driver Attributes

Table 37 shows the attributes for the Chrontel CH7307* port driver.

Note: For flat panel backlight timing settings, please see Table 10 in Section 3.0.

Table 37. Chrontel CH7307 Port Driver Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
Spread Spectrum Clocking	43	Spectrum clocking	0-15 Default = 0 Step = 1
FIXED TIMING	60	This indicates whether attached display is a fixed timing display.	0 = on 1 = off

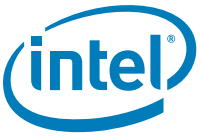
B.2.7 Chrontel CH7308 Port Driver Attributes

Table 38 shows the attributes for the Chrontel CH7308* port driver.

Note: For FPINFO panel width, height, and backlight timing settings, please see Table 10 in Section 3.0.

Table 38. Chrontel CH7308 Port Driver Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
LVDS Color Depth	26	Panel depth	18 = 18 bits 24 = 24 bits Default = 18
DUAL_CHANNEL	27	Dual-channel pane	Default - 0
Spread Spectrum Clocking	43	Spectrum Clocking	0-15 Default = 0 Step = 1
Dither	45	Dither setting	Default = 0
HSync Panel Protection	46	Horizontal sync panel protection	Default = 0
VSync Panel Protection	47	Vertical sync panel protection	Default = 0
Pixel Clock Protection	48	Pixel clock protection	Default = 0
LVDS Panel Type	49	LVDS panel connector.	0 = SPWG 1 = OpenLDI Default = 0
Text Enhancement	58	Controls text tuning.	0-4.
Fixed Timing	60	This indicates whether attached display is a fixed timing display.	0 = on 1 = off



B.2.8 Chrontel CH7315/CH7319/CH7320 Port Driver Attributes

Table 39 shows the attributes for the Chrontel CH7315/CH7319/CH7320 port driver.

Table 39. Chrontel CH7315/CH7319/CH7320 Port Driver Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
Fixed Timing	60	This indicates whether attached display is a fixed timing display.	0 = on 1 = off

B.2.9 Focus FS453/FS454 Port Driver TV Attributes

Table 40 lists the TV attributes for the Focus FS453* and FS454* port drivers.

Table 40. Focus FS453/FS454 Port Driver TV Attributes (Sheet 1 of 2)

Attribute Name	Attribute ID	Description	Possible Ranges
BRIGHTNESS	0	Screen Brightness	0-100
CONTRAST	1	Color Contrast	0-100
FLICKER	3	Flicker Filter	0-1000, default is 1000
HPOSITION	4	Horizontal Position	0-100
VPOSITION	5	Vertical Position	0-100
HSCALE	6	Horizontal Scale	0-1000, increment by 10.
VSCALE	7	Vertical Scale	0-1000, increment by 10.
TVFORMAT	8	TV Format	1 = NTSC 2 = NTSC-EIAJ 3 = PAL 4 = PAL-D 5 = PAL-G 6 = PAL-H 7 = PAL-I 8 = PAL-M 9 = PAL-N 10 = PAL-NC 11 = PAL-60 12 = 480P 13 = 720P
LUMAFILTER	10	Luma Filter	0 = Disabled 1 = Enabled
CHROMAFILTER	11	Chroma Filter	0 = Disabled 1 = Enabled
TVOUTTYPE	14	TV Output Format	1 = CVBS 2 = CVBS+Y/C 3 = Y/C 4 = YPrPb 5 = SCART-RGB 6 = VGA-DAC Note: TV Output Types are limited to S-Video and Composite for the VBIOS.
SATURATION	15	Color Saturation	0-100. Default is 50.

Table 40. Focus FS453/FS454 Port Driver TV Attributes (Continued) (Sheet 2 of 2)

Attribute Name	Attribute ID	Description	Possible Ranges
REVERSE DVO DATA	30	Reverses the data order coming out of the DVO port.	0 = Normal DVO data order 1 = Reverse the DVO RGB data order. Default is 0.
SHARPNESS	31	Image Sharpness	0-1000. Default is 800.
HWCONFIG	32	Hardware Configuration. This attribute allows you to select a hardware configuration for the FS453/FS454 port drivers.	0 = Standard configuration 1 = FS454 ADD card 2 = ADD card (Japan J connector) 3 = Standard configuration (invert clock) Default is 3 for the FS454.
HORZFILTER	33	Horizontal Filter	0-100. Default is 50.

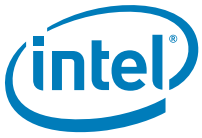
B.2.10 National Semiconductor NS387R Port Driver LVDS Attributes

Table 41 lists the LVDS attributes for the National Semiconductor NS387R* port driver.

Note: For flat panel backlight timing settings, please see Table 10 in Section 3.0.

Table 41. National Semiconductor NS387R Port Driver LVDS Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
GANG MODE	28	For achieving a Gang mode output using two digital ports.	0 = Off 1 = On Default = 0
GANG MODE EVEN ODD	29	Gang display even or odd. This attribute is to be set along with Gang mode (28). This mode (Gang Mode Even Odd) puts even pixels on one digital port and odd pixels on the other, and needs to be selected based on the display panel used.	0 = Normal mode (half pixel on one port, remaining half on another port) 1 = Even-Odd mode (even pixels on one port, odd pixels on another port). Default = 0
REVERSE DVO DATA	30	Reverses the data order coming out of the DVO port.	0 = Normal DVO data order 1 = Reverse the DVO RGB data order. Default is 0.
GANG DVO CLOCK INVERSION	56	Controls DVO clock inversion in Gang mode. This attribute is to be set along with the "Gang Mode" attribute (28).	0 = No clock inversion 1 = DVO clock inversion. Default is 0.
VGA 2X IMAGE	57	Controls VGA image in Gang mode.	0 = No VGA 2x image 1 = VGA 2x image. Default is 1.
FIXED TIMING	60	This indicates whether attached display is a fixed timing display.	0 = on 1 = off



B.2.11 National Semiconductor NS2501 Port Driver Attributes

Table 42 shows the attributes for the National NS2501.

Table 42. National Semiconductor NS2501 Port Driver Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
FP WIDTH	32769	Attached panel width	Supported values are 800, 1024, 1280, 1400. Default value is 1280.
FP HEIGHT	32770	Attached panel height	Default - 0
Spread Spectrum Clocking	43	Spectrum Clocking	Supported values are 600, 768, 768, 1050. Default value is 768.

B.2.12 Silicon Image SiI 164 Port Driver DVI Attributes

Table 43 lists the DVI attributes for the Silicon Image SiI 164* port driver.

Note: For flat panel backlight timing settings, please see Table 10 in Section 3.0.

Table 43. Silicon Image SiI 164 Port Driver DVI Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
FIXED TIMING	60	This indicates whether attached display is a fixed timing display.	0 = on 1 = off
EDGE SELECT	32769	Allows you to invert the input clock signal.	0 = Input data is falling edge latched 1 = Input data is rising edge latched Default = 0

B.2.13 Silicon Image SiI 1362/SiI 1364 Port Driver DVI Attributes

Note: For flat panel backlight timing settings, please see Table 10 in Section 3.0.

Table 44. Silicon Image SiI 1362/SiI 1364 Port Driver DVI Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
FIXED TIMING	60	This indicates whether attached display is a fixed timing display.	0 = on 1 = off

B.2.14 Texas Instruments TFP410 DVI Port Driver Attributes

The Texas Instruments TFP410* transmitter is supported on the following chipsets:

- Intel® 852GM chipset
- Intel® 852GME chipset
- Intel® 855GME chipset

Table 45 lists the DVI attributes for the Texas Instruments TFP410 port drivers.

Note: For flat panel backlight timing settings, please see Table 10 in Section 3.0.

Table 45. Texas Instruments TFP 410 DVI Port Driver Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
FIXED TIMING	60	This indicates whether attached display is a fixed timing display.	0 = on 1 = off
EDGE SELECT	32771	Allows you to invert the input clock signal.	0 = Input data is falling edge latched 1 = Input data is rising edge latched Default = 0
BUS SELECT	32771	Allows you to select between 12-bit dual edge mode (value = 0) or 4-bit single edge mode (value=1).	0 = 12-bit dual edge mode 1 = 4-bit single edge mode Default = 1

B.3 Chipset and Port Driver-Specific Installation Information

Table 46. Default Search Order

Chipset	Default Search Order
Intel® Q35	ANALOG, sDVOB, sDVOC
Intel® GME965	ANALOG, sDVOB, sDVOC, LVDS
Intel® Q965	ANALOG, sDVOB, sDVOC
Intel® 945GME	ANALOG, sDVOB, sDVOC, LVDS
Intel® 945G	ANALOG, sDVOB, sDVOC
Intel® 915GV	ANALOG, sDVOB, sDVOC
Intel® 915GME	ANALOG, sDVOB, sDVOC, LVDS
Intel® 910GMLE	ANALOG, sDVOB, sDVOC, LVDS
Intel® 855GME	ANALOG, DVOC, DVOB, LVDS
Intel® 852GME	ANALOG, DVOC, DVOB, LVDS
Intel® 852GM	ANALOG, DVOC, DVOB, LVDS

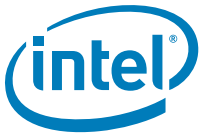


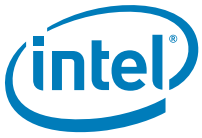
Table 47. Default GPIO Pin Pair Assignments

Chipset	Default GPIO Pin Pair for EDID			
	DVO/A	DVOB	DVOC	LVDS
Intel® Q35	N/A	4	4	N/A
Intel® GME965	N/A	4	4	2
Intel® Q965	N/A	4	4	N/A
Intel® 945GME	N/A	4	4	2
Intel® 945G	N/A	4	4	N/A
Intel® 915GV	N/A	4	4	N/A
Intel® 915GM	N/A	4	4	2
Intel® 910GMLE	N/A	4	4	2
Intel® 855GME	2	3	3	2
Intel® 852GME	2	3	3	2
Intel® 852GM	2	3	3	2

Table 48. Default I^2C Device Address Byte Assignment

Port Driver	Default Device Address Bytes (DAB)
CH7021, CH7315, CH7317, CH7319, CH7320	0x70 (for first sDVO device) 0x72 (for second sDVO device)
CH7009	0xEC and then tries at 0xEA
CH7307	0x70 (for first sDVO device) 0x72 (for second sDVO device)
CH7308	0x70 (for first sDVO device) 0x72 (for second sDVO device)
FS454	0x94 and then tries at 0xD4
NS2501	0x70
NS387R	0x70
SiI 164	0x70
SiI 1362	0x70 (for first sDVO device) 0x72 (for second sDVO device)
SiI 1364	0x70 (for first sDVO device) 0x72 (for second sDVO device)
TFP410	0x70
TH164	0x70

This page is intentionally left blank.



Appendix C Intel® 5F Extended Interface Functions

The BIOS provides a set of proprietary function calls to control operation of the extended features. These function calls all use AH = 5Fh in their designed interface for easy identification as a proprietary function

These functions are designed to maintain maximum compatibility with the Desktop and Mobile Video BIOS. As such many of the definitions behave identically. When the behavior of the Embedded Video BIOS is not identical to the Desktop and Mobile Video BIOS it is noted.

In addition to these 5F functions, the Video BIOS also supports all 4F functions defined by the *VESA BIOS Extension (VBE) Core Functions Standard, Version 3.0* with the exception of the 0A function (Return VBE Protected Mode Interface). All other functions, from 00 through 09 and 0B are supported by the Video BIOS. Click on the following link to view the VBE 3.0 Core Functions Standard document.

http://www.vesa.org/Public/VBE/vbe3.pdf

Table 49 provides a summary of the IEGD supported Intel 5F functions.

Table 49. Summary of Intel 5F Extended Interface Functions

Function	Function Name	Description
BIOS Extended Interface Functions		
5F01h	Get Video BIOS Information	Gets VBIOS Build Information.
5F05h	Refresh Rate	Sets a new vertical refresh rate for a given mode and returns the current vertical refresh rate
5F10h	Get Display Memory Information	Returns information about the linear memory.
5F1Ch	BIOS Pipe Access	Sets the BIOS pipe access and returns the BIOS pipe access status.
5F29h	Get Mode Information	Returns information on the requested mode.
5F61h	Local Flat Panel Support Function	Supports local flat panel features.
Hooks for the System BIOS		
5F31h	POST Completion Notification Hook	Signals the completion of video POST (Power On Self Test)
5F33h	Hook After Mode Set	Allows System BIOS to intercept Video BIOS at the end of a mode set.
5F35h	Boot Display Device Hook	Allows System BIOS to override video display default setting.
5F36h	Boot TV Format Hook	Allows System BIOS to boot TV in selected TV format state.
5F38h	Hook Before Set Mode	Allows System BIOS to intercept Video BIOS before setting the mode.
5F40h	Config ID Hook	Allows System BIOS to supply a configuration ID that is passed to the driver.

C.1 BIOS Extended Interface Functions

The BIOS provides a set of proprietary function calls to control operation of the extended features. These function calls all use AH = 5Fh in their designed interface for easy identification as a proprietary function

These functions are designed to maintain maximum compatibility with the Desktop and Mobile Video BIOS. As such many of the definitions behave identically. When the behavior of the Embedded Video BIOS is not identical to the Desktop and Mobile Video BIOS it is noted.

C.1.1 5F01h – Get Video BIOS Information

This function returns the Video BIOS Build information.

Note: This function is an extension of the Desktop and Mobile Video BIOS. If register ECX does not contain ASCII characters “IEGD” then the VBIOS is not described by this specification.

Calling Register:

AX = 5F01h, Get Video Information function

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
EBX = 4 bytes Video BIOS Build Number ASCII string, e.g. ‘1000’
ECX = 4 bytes Embedded Identifier, ASCII string ‘IEGD’

C.1.2 5F05h – Refresh Rate

This function sets a new vertical refresh rate for a given mode and returns the current vertical refresh rate and available refresh rate for a given non-VGA mode.

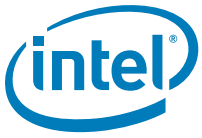
C.1.2.1 5F05h, 00h – Set Refresh Rate

This sub-function sets a new default refresh rate for the selected pipe. If the mode is currently active, the CRT controller and other registers will be automatically programmed setting the requested refresh rate.

Note: This function is not entirely compatible with the Desktop and Mobile versions. It is not possible to set the refresh rate for a given mode in advance. This function sets the “desired” refresh rate which will be applied to all subsequent mode sets when possible. If the mode provided in BL is the current mode, then a mode change will be automatically performed.

Calling Register:

AX = 5F05h, Refresh Rate function
BH = 00h, Set Refresh Rate sub-function
BL = Mode Number
ECX = Refresh rate (indicated by setting one bit):
Bits 31 - 9= Reserved
Bit 8= 120 Hz
Bit 7= 100 Hz
Bit 6= 85 Hz
Bit 5= 75 Hz
Bit 4= 72 Hz
Bit 3= 70 Hz



Bit 2= 60 Hz
Bit 1= 56 Hz
Bit 0= 43 Hz (Interlaced - Not supported)

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed

C.1.2.2 5F05h, 01h – Get Refresh Rate

This sub-function returns current vertical refresh rate for the selected pipe and available refresh rates information for a given Non-VGA mode.

Note: This sub-function returns a status of supported but failed (AX = 015Fh) if executed with a standard VGA mode.

Calling Registers:

AX = 5F05h, Refresh Rate function
BH = 01h, Get Refresh Rate sub-function
BL = Mode number

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
EBX = Available refresh rates (indicated by one or more bits set):
Bits 31 - 9 = Reserved
Bit 8= 120 Hz
Bit 7= 100 Hz
Bit 6= 85 Hz
Bit 5= 75 Hz
Bit 4= 72 Hz
Bit 3= 70 Hz
Bit 2= 60 Hz
Bit 1= 56 Hz
Bit 0= 43 Hz (Interlaced - Not supported)
ECX = Current refresh rate (see EBX for bit definitions)

C.1.3 5F10h – Get Display Memory Information

This function returns information regarding the linear memory starting address, size and memory mapped base address.

Calling Register:

AX = 5F10h, Get Linear Display Memory Information function

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
ESI = Display memory base address
ECX= Total physical display memory size (in bytes)
EDX= Available display memory size (in bytes)
EDI = Memory Mapped I/O Base Address
EBX= Stride (memory scan line width in bytes)

C.1.4 5F1Ch – BIOS Pipe Access

This function will set the BIOS pipe access or return the BIOS pipe access status.

C.1.4.1 5F1Ch, 00h – Set BIOS Pipe Access

This sub-function will set the currently selected pipe. All 5f functions operate on the currently selected pipe.

When not in clone modes this value cannot be set.

Calling Registers:

AX = 5F1Ch, BIOS Pipe Access function
BH = 00h, Set BIOS Pipe Access sub-function
CH = BIOS Pipe access:
= 00h, Pipe A
= 01h, Pipe B

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed

C.1.4.2 5F1Ch, 01h – Get BIOS Pipe Access

This sub-function will return the currently selected pipe.

Calling Registers:

AX = 5F1Ch, BIOS Pipe Access function
BH = 01h, Get BIOS Pipe Access sub-function

Return Registers:

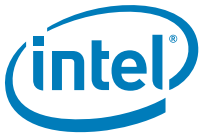
AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
CH = BIOS Pipe access:
= 00h, Pipe A
= 01h, Pipe B

C.1.5 5F29h – Get Mode Information

This function returns the requested mode's resolution, color depth, and maximum required bandwidth using its current refresh rate. This function is applied to extended-graphics modes only. If the mode number is not an extended graphics mode, the function will return failure.

Calling Registers:

AX = 5F29h, Get Mode Information function
BH = Mode To Use:
= 80h, Current Mode
= 00h - 7Fh, Given Mode Number



Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
EBX bits 31 - 16= Mode horizontal (X) resolution in pixels
EBX bits 15 - 0= Mode vertical (Y) resolution in pixels
ECX bits 31 - 16= Maximum bandwidth in megabytes per second
ECX bits 15 - 0= Color depth in bits per pixel

C.1.6 5F61h – Local Flat Panel Support Function

This function supports local flat panel only features.

Note: Only Subfunction 5h of the 5f61h interface is supported for the Embedded vBIOS.

C.1.6.1 5F61h, 05h – Get Configuration ID

This function is used to return the Configuration ID.

Note: This function is known as “Get Local Flat Panel Number” in the Desktop and Mobile Video BIOS. This function performs a similar purpose however, the configuration IDs have no pre-defined meaning. The Configuration ID is reported to the Embedded Graphics Driver and will be used as described in the *Intel® Embedded Graphics Drivers and Video BIOS User’s Guide*.

Calling Registers:

AX = 5F61h, Local Flat Panel Support function
BH = 05h, Get Config ID Subfunction

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
BL = Config ID

C.1.7 5F68h – System BIOS Callback

This is a generic function that allows SoftBIOS to do any system callbacks through INT 15h. The Input/Output of this function is dependent on the definition of the desired INT 15h hook except for the EAX register.

Calling Registers:

AX = 5F68h, System BIOS Callback Function
EAX bits 31:16= System BIOS INT 15h Hook Function

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed

C.2 Hooks for the System BIOS

The video BIOS performs several system BIOS interrupt function calls (interrupt 15h hooks). Each function provides the system BIOS with the opportunity to gain control at specific times to perform any custom processing that may be required. After each interrupt hook, the system BIOS must return control to the video BIOS. INT 10h calls could be made within the INT 15h hook calls provided that it is not recursive and thus cause a deadlock.

C.2.1 5F31h – POST Completion Notification Hook

This hook signals the completion of video POST (Power On Self Test). The hook executes after the sign-on message is displayed and PCI BIOS resizing.

Calling Registers:

AX = 5F31h, POST Completion Notification Hook

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 015Fh, Function supported but failed
= 005Fh, Function supported and successful

C.2.2 5F33h – Hook After Mode Set

This hook allows the system BIOS to intercept the video BIOS at the end of a mode set.

Calling Registers:

AX = 5F33h, Hook After Mode Set
BH = Number of character columns
BL = Current mode number
CH = Active display page

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 015Fh, Function supported but failed
= 005Fh, Function supported and successful

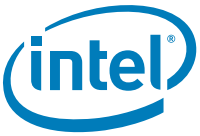
C.2.3 5F35h – Boot Display Device Hook

This hook allows the system BIOS to override the video display default setting. The graphics BIOS will set the returned video display during POST (power up initialization).

Note: This function is not entirely compatible with the Desktop and Mobile Video BIOS. The bits in CL have a configurable mapping to the Port Numbers as defined in the *Intel® Embedded Graphics Drivers and Video BIOS User's Guide*. The assigned meanings used in the Desktop specification can be duplicated with a correct configuration. The values below are the default values if no "Common To Port" mapping is provided.

Calling Registers:

AX = 5F35h, Boot Display Device Hook



Return Registers:

AX = Return Status (function not supported if AL != 5Fh);
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
CL = Display Device Combination to boot (1 = Enable display, 0 = Disable display):
= 00h, VBIOS Default
Bit 7 - 6= Reserved
Bit 5= Port 5 (or common_to_port[5])
Bit 4= Port 4 (or common_to_port[4])
Bit 3= Port 3 (or common_to_port[3])
Bit 2= Port 2 (or common_to_port[2])
Bit 1= Port 1 (or common_to_port[1])
Bit 0= Port 0 (or common_to_port[0])

C.2.4 5F36h – Boot TV Format Hook

This hook allows the system BIOS to boot TV in selected TV format state.

Calling Registers:

AX = 5F36h, Boot TV Format Hook

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 015Fh, Function supported but failed
= 005Fh, Function supported and successful
BL = TV Format requested:
= 00h, No Preference
= 01h, NTSC_M
= 11h, NTSC_M_J
= 21h, NTSC_433
= 31h, NTSC_N
= 02h, PAL_B
= 12h, PAL_G
= 22h, PAL_D
= 32h, PAL_H
= 42h, PAL_I
= 52h, PAL_M
= 62h, PAL_N
= 72h, PAL_60
= 03h, SECAM_L
= 13h, SECAM_L1
= 23h, SECAM_B
= 33h, SECAM_D
= 43h, SECAM_G
= 53h, SECAM_H
= 63h, SECAM_K
= 73h, SECAM_K1

C.2.5 5F38h – Hook Before Set Mode

This hook allows the system BIOS to intercept the video BIOS before setting the mode.

Calling Registers:

AX = 5F38h, Hook Before Set Mode
CL = New video mode to be set

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 015Fh, Function supported but failed
= 005Fh, Function supported and successful

C.2.6 5F40h – Config ID Hook

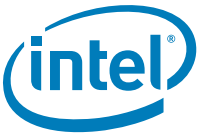
This function is known as “Boot Panel Type Hook” in the Desktop and Mobile Video BIOS. It allows the system BIOS to supply a configuration ID that will eventually be passed to the driver. This configuration ID is unused by the Video BIOS; however, it alters the behavior of the driver as described in the *Intel® Embedded Graphics Drivers and Video BIOS User’s Guide*.

Calling Registers:

AX = 5F40h, Config ID Hook

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
CL = Configuration ID



Appendix D Intel® OpenGL APIs

Table 50 presents **supported** IEGD OpenGL* APIs, and Table 51 presents **non-supported** IEGD OpenGL APIs.

The following OpenGL versions are supported:

- Version 1.3 on all Embedded Intel® Architecture (eIA) chipsets
- Version 1.4 on 915GV, 915GM, 945G, 945GM, Q965, GME965
- Version 1.5 on Q965, GME965

For general OpenGL information, visit http://www.opengl.org/about/overview/

Table 50. Supported Intel® OpenGL APIs (Sheet 1 of 2)

Supported API Name(s)
GL_3DFX_texture_compression_FXT
GL_ARB_depth_texture[1]
GL_ARB_fragment_program (965 only)
GL_ARB_multitexture
GL_ARB_occlusion_query (956 only)
GL_ARB_shadow[1]
GL_ARB_textture_env_dot3
GL_ARB_texture_border_clamp
GL_ARB_texture_compression
GL_ARB_texture_cube_map
GL_ARB_texture_env_add
GL_ARB_texture_env_combine
GL_ARB_texture_env_crossbar
GL_ARB_transpose_matrix
GL_ARB_vertex_buffer_object
GL_ARB_vertex_program (965 only)
GL_EXT_abgr
GL_EXT_bgra
GL_EXT_blend_color
GL_EXT_blend_func_separate
GL_EXT_blend_minmax
GL_EXT_blend_subtract
GL_EXT_clip_volume_hint
[1] Only supported on Intel 915 series and later chipsets.

Table 50. Supported Intel® OpenGL APIs (Sheet 2 of 2)

Supported API Name(s)
GL_EXT_compiled_vertex_array
GL_EXT_cull_vertex
GL_EXT_fog_coord
GL_EXT_mulit_draw_arrays
GL_EXT_packed_pixels
GL_EXT_rescale_normal
GL_EXT_secondary_color
GL_EXT_separate_specular_color
GL_EXT_shadow_funcs[1]
GL_EXT_stencil_two_side[1]
GL_EXT_texture_compression_s3tc
GL_EXT_texture_env_add
GL_EXT_texture_filter_anisotropic
GL_EXT_texture_lod_bias (965 only)
GL_IBM_texture_mirrored_repeat
GL_NV_blend_square
GLX_ARB_get_proc_address
[1] Only supported on Intel 915 series and later chipsets.

Table 51. Non-Supported Intel® OpenGL APIs

Non-Supported API Name(s)
GL_ARB_color_buffer_float
GL_ARB_fragment_program_shadow
GL_ARB_point_sprite
GL_ARB_shader_objects
GL_ARB_shading_language_100
GL_ARB_texture_non_power_of_two
GL_EXT_paletted_texture
GL_WIN_swap_hint
WGL_ARB_buffer_region
WGL_ARB_extensions_string
WGL_ARB_make_current_read
WGL_ARB_pbuffer
WGL_ARB_pixel_format
WGL_EXT_swap_control